

MAGAZINE

# BSD

FOR NOVICE AND ADVANCED USERS

**FIRST LOOK AT THE RENEWED CTL HIGH  
AVAILABILITY IMPLEMENTATION IN FREEBSD**

CREATE YOUR FIRST  
**FREEBSD KERNEL MODULE**

SECURING THE FUTURE OF THE WEB

**BHYVE: INTRODUCTION TO HYPERVISORS**

HOW TO INSTALL **ODOO ERP SOFTWARE**  
ON **UBUNTU 16.04**

**MEETBSD 2016 REPORT**

VOL 10 NO 11

ISSUE 10/2016 (87)

1898-9144



## Dear Readers,

Christmas time is fast approaching, so we all need to get busy with shopping. If you are looking for a perfect gift for your techy friend, we would like to suggest our new online course, DevOps with Chef on FreeBSD. This training course imparts knowledge on the tools, best practices, and skills to automate your FreeBSD servers. Our training is loaded with practical, real-world tools and techniques. Upon completion, the experience will enable you to implement DevOps in your IT projects almost immediately.

As Benjamin Franklin once said, "an investment in knowledge pays the best interest." In that spirit, we are happy to deliver another repository of knowledge this month, starting with another great article, "First Look at the Renewed CTL High Availability Implementation in FreeBSD" by Mikhail Zakharov. Also in the FreeBSD Corner, Abdelhadi Khiati will explain how to "Create Your First FreeBSD Kernel Module." This article will serve as an introduction to an article by Abdelhadi which will be in next month's issue.

Moving on to the security section, Brian Spector, CEO of MIRACL, shared great information with us through his insightful article about "Securing the Future of the Web".

Bhyve has been the most requested topic from prospective students. Abdorrahman Homaei has put together a great piece on the subject, "bhyve: Introduction to Hypervisors." We hope you enjoy it!

Next, we'll post an article about, "How to Install Odoo ERP Software on Ubuntu 16.04," by Moustafa Nabil El-Zeny. We know it's not about BSD, but we hope you will enjoy the tutorial regardless. It's an amazing, in-depth piece about Odoo which has never been covered by BSD Mag.

In case you didn't make it for MeetBSD held at U.C. Berkeley in California, Michael Dexter shared his "MeetBSD 2016 Report". In a nutshell, it was a wonderful feeling to be a part of the event.

Finally, you will find Rob's Column. This time he is sharing information about cyber fraud in the banking industry.

We hope you will enjoy the issue as we approach the cold days of winter (at least in some parts of the world). Here, in Denmark, we had snow at the beginning of November, which was very unusual. Winter and short cold days are coming, thus, we would like to encourage you to once again take a look at our new online course by clicking here >>

<https://bsdmag.org/course/devops-chef-freebsd/>.

Rest assured, the course will be a perfect filling for the long evenings.

**Marta & BSD Team**

# MAGAZINE BSD

## Editor in Chief:

Marta Ziemianowicz

[marta.ziemianowicz@software.com.pl](mailto:marta.ziemianowicz@software.com.pl)

## Contributing:

Mikhail Zakharov, Abdelhadi Khiati, Mike Tenesca, Brian Spector, Abdorrahman Homaei, Moustafa Nabil El-Zeny, Michael Dexter and Rob Somerville.

## Top Betatesters & Proofreaders:

Denise Ebery, Eric Geissinger, Luca Ferrari, Imad Soltani, Olaoluwa Omo-kanwaye, Radjis Mahangoe, Mani Kanth and Mark VonFange.

## Special Thanks:

Annie Zhang

Denise Ebery

## DTP:

Marta Ziemianowicz

## Senior Consultant/Publisher:

Paweł Marciniak

[pawel@software.com.pl](mailto:pawel@software.com.pl)

## CEO:

Joanna Kretowicz

[joanna.kretowicz@software.com.pl](mailto:joanna.kretowicz@software.com.pl)

## Publisher:

Hakin9 Media SK 02-676 Warsaw, Poland Postępu 17D Poland worldwide  
[publishing@bsdmag.org](mailto:publishing@bsdmag.org) [www.bsdmag.org](http://www.bsdmag.org)

Hakin9 Media SK is looking for partners from all over the world. If you are interested in cooperation with us, please contact us via e-mail:  
[editors@bsdmag.org](mailto:editors@bsdmag.org)

All trademarks presented in the magazine were used only for informative purposes. All rights to trademarks presented in the magazine are reserved by the companies which own them.



## News

### [BSD World Monthly News](#) **4**

*by Marta Ziemianowicz*

This column presents the latest news coverage of events, product releases and trending topics.

## FreeBSD Corner

### [First Look at the Renewed CTL High Availability Implementation in FreeBSD](#) **14**

*by Mikhail Zakharov*

This enhancement looks extremely important for the BeaST storage system, as implementation of high available native ALUA in FreeBSD can potentially replace the BeaST arbitration mechanism (“Arbitrator”), which is completely described in the papers on the BeaST project page.

### [Create Your First FreeBSD Kernel Module](#) **33**

*by Abdelhadi Khiati*

Security on the Internet is long overdue for change. Whether it's by script kiddies or state actors, the Internet has shown that it can be attacked in a myriad of new and inventive ways. This poses huge risks to the future of our digital economy: nearly five billion private data records have been exposed globally since 2013, and barely a week goes by without a new data breach or vulnerability being revealed.

## Security

### [Securing the Future of the Web](#) **38**

*by Brian Spector, CEO at MIRACL*

Security on the Internet is long overdue for change. Whether it's by script kiddies or state actors, the Internet has shown that it can be attacked in a myriad of new and inventive ways. This poses huge risks to the future of our digital economy: nearly five billion private data records have been exposed globally since 2013, and barely a week goes by without a new data breach or vulnerability being revealed.

## bhyve

### [bhyve: Introduction to Hypervisors](#) **43**

*by Abdorrahman Homaei*

A hypervisor or virtual machine monitor (VMM) is a computer software, firmware or hardware that creates and runs virtual machines. Actually, the power of VMM depends on the kernel model of the operating system. In general, there are three types of kernel model, microkernel, monolithic and hybrid. Here are pros and cons to each type.

## Ubuntu

### [How to install Odoo ERP Software on Ubuntu 16.04](#) **57**

*by Moustafa N. El-Zeny*

Odoo is a web-based Open Source enterprise resource planning and customer relationship software that helps to organize and grow your business. Odoo was formerly known as OpenERP and, before that, TinyERP. There are many apps available to extend Odoo, for example: billing, accounting, manufacturing, purchasing, warehouse management, and project Management.

## MeetBSD

### [MeetBSD 2016 Report](#) **76**

*by Michael Dexter*

### [Rob's Column](#) **79**

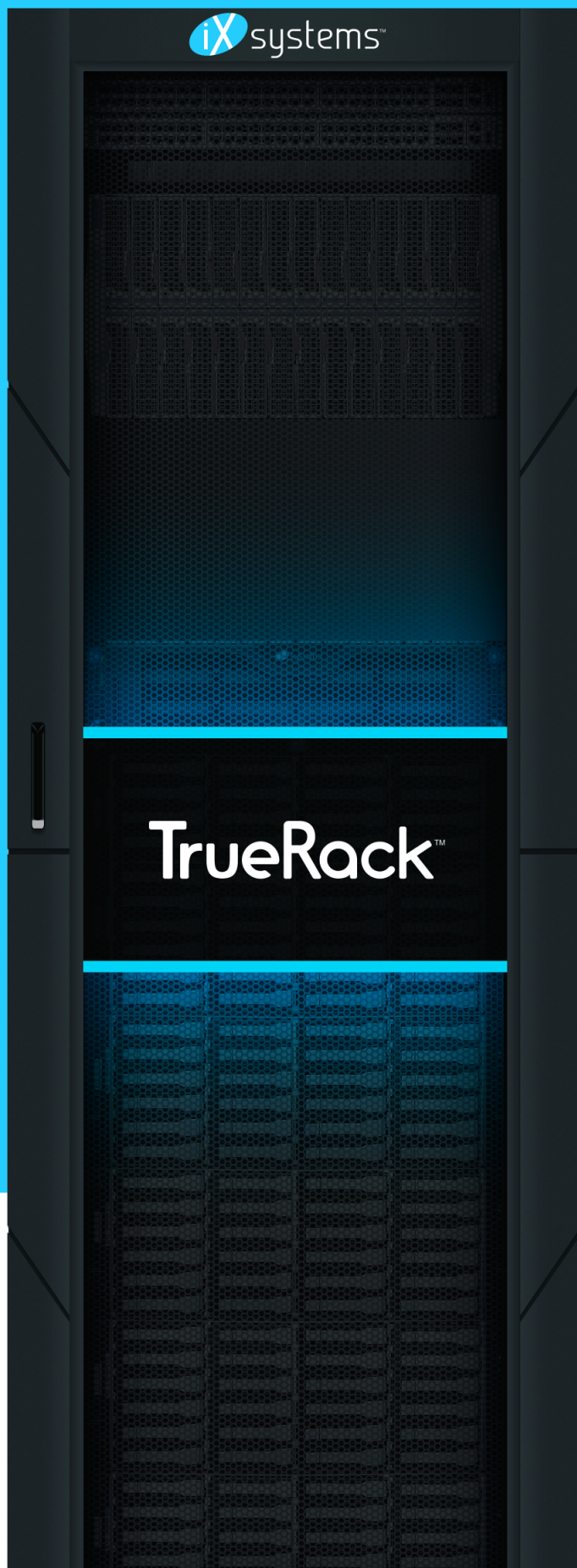
*by Rob Somerville*

Over the weekend, Tesco Bank suspended online transactions after an attacker gained access to over 20,000 accounts, with money being withdrawn fraudulently in some cases. In another security incident, the City of El Paso was the victim of CEO fraud worth over \$3.2 million. What implications does this paradigm shift towards online crime have for us as a society?



# Simplify your Data Center

**Meet TrueRack™** — A powerfully flexible rack-scale architecture that takes the guesswork out of building large scale data center applications.



- Converged Infrastructure
- Customizable for Virtualization, Big Data, Cloud & Hyperscale
- Up to 70% Lower TCO Than AWS
- Scalable and Repeatable Deployments

For more information on TrueRack, visit **[ixsystems.com/TrueRack](https://ixsystems.com/TrueRack)** today



## BSD Certification

**The BSD Certification Group Inc. (BSDCG) is a non-profit organization committed to creating and maintaining a global certification standard for system administration on BSD based operating systems.**

### ? WHAT CERTIFICATIONS ARE AVAILABLE?

**BSDA: Entry-level certification** suited for candidates with a general Unix background and at least six months of experience with BSD systems.

**BDSP: Advanced certification** for senior system administrators with at least three years of experience on BSD systems. Successful BDSP candidates are able to demonstrate strong to expert skills in BSD Unix system administration.

### ✓ WHERE CAN I GET CERTIFIED?

We're pleased to announce that after 7 months of negotiations and the work required to make the exam available in a computer based format, that the BSDA exam is now available at several hundred testing centers around the world. Paper based BSDA exams cost \$75 USD. Computer based BSDA exams cost \$150 USD. The price of the BDSP exams are yet to be determined.

Payments are made through our registration website:  
<https://register.bsdcertification.org/register/payment>

### i WHERE CAN I GET MORE INFORMATION?

More information and links to our mailing lists, LinkedIn groups, and Facebook group are available at our website:  
<http://www.bsdcertification.org>

Registration for upcoming exam events is available at our registration website:  
<https://register.bsdcertification.org/register/get-a-bsdcg-id>

## Amazon's very own Linux is now available for download.

**It turns out you need to test on-premises before you send an app to the cloud.**

Amazon Web Services is letting customers download its own artisanal Linux.

The company has loosed its Linux Container Image to assist those planning a move to its cloud test their software and workloads on-premises.

Previously, the image was only accessible on-cloud, for customers running virtual machine instances on AWS.

The cloud giant's chief evangelist, Jeff Barr, made the announcement in this blog post.

Barr notes that the Linux config is designed for security: there's no remote root access; SSH only runs using key pairs, not passwords; and it's built with a very small number of "non-critical packages.

"It is built from the same source code and packages as the [Amazon Machine Image] AMI and will give you a smooth path to container adoption. You can use it as-is or as the basis for your images," Barr notes.

One thing to note, on AWS, Amazon handles the business of updating critical or important security updates at boot; as an on-premises instance, users will have to run their patches.

[http://www.theregister.co.uk/2016/11/03/now\\_you\\_can\\_run\\_the\\_same\\_linux\\_as\\_aws\\_at\\_home/](http://www.theregister.co.uk/2016/11/03/now_you_can_run_the_same_linux_as_aws_at_home/)



## Raspberry Pi continues to blaze new trails



This post consists of the highlights of the exhilarating talk I gave at the conference, which covered the mission and purpose of Raspberry Pi and our programs and outreach including Code Club for kids, Raspberry Jams, and Picademy.

I'm Raspberry Pi's Community Manager, based in Cambridge, UK. We have sold over 10 million Raspberry Pis since launching our first product in

2012. Many people think of Raspberry Pi as a hardware company, but in fact, we're an education charity.

In 2006, the Raspberry Pi Foundation was founded at Cambridge University with the modest goal of getting more people to study computer science. The proposed solution, a small and cheap Linux computer, was immediately familiar with a large number of individuals. Since 2012, educators, hobbyists and industrial users have been making the most of our range of devices and funding our education programs. All profits made by our trading subsidiary go to the Foundation.

### **Raspberry Pi Foundation's mission**

Our mission is putting the power of digital making into the hands of people all over the world. We do this by providing low-cost, high-performance computers that people use to learn, solve problems and have fun. Moreover, our outreach and educational awareness helps more people access computing and digital making. We develop free resources not only to help people learn about computing but also how to make things with computers. We also train educators to guide other people to learn.



### **Current Raspberry Pi models**

The Pi 3 is our headline product. It's a 64-bit with 1GB RAM, which is not just a good computer for \$35, it's a good computer, full stop.

The Pi Zero shook the world last year when we released it for just \$5 and stuck a free one on the front of our print magazine, The MagPi.

If you used a Pi in 2012, it might have felt a bit sluggish and looked a little ugly. Currently, it runs fast and looks quite appealing. We just released the new Raspbian (our distro based



<div>Raspberry Pi 3</div> <div></div>	<div>Raspberry Pi Zero</div> <div></div>
64-bit quad-core ARMv8 @ 1.2GHz	32-bit single-core ARMv6 @ 900MHz
1GB RAM	512MB RAM
\$35	\$5

on Debian) with a new desktop called PIXEL.

### Free resources and training

We provide a set of free learning resources on our website. There are plenty of fun activities of things you can do with a Pi. They're free in every sense of the word and you can contribute on GitHub.

Picademy is our free teacher training program that we run in

the UK and the US. Educators apply for a place to become Raspberry Pi certified educators and become part of a global community.

### Community

#### Raspberry Jam

Raspberry Jams are community events set up around learning and sharing with the Pi. They're family-friendly meetups and anyone can start their own. Have a look at our map and if there isn't one near you, get in touch and I'll help you get started.

### Code Club

Code Club is a network of after-school coding clubs for kids aged between 9 to 11 years. Anyone can set up a Code Club, and we provide training, support, and resources. Check out the Code Club World website for more information.

<https://opensource.com/life/16/11/raspberry-pi-continues-blaze-new-trails>



## Snappy Ubuntu Core 16 Launched For IoT Devices, Raspberry Pi, and Cloud With A Focus On Security

**Ubuntu Core 16 is a minimalistic Linux-based operating system developed primarily for IoT devices. The latest Snappy Ubuntu borrows core features from the Ubuntu 16.10 Yakkety Yak and takes advantage of the Snaps.**

The Linux distro for IoT devices takes advantage of the Snap packages — a zip file containing all the package data along with the details of running the application on the machine. The Snaps are tamper-proof, developer-friendly and digitally signed. The Snappy package manager used to install and manage snaps has been developed by Canonical itself and it enhances security by enabling sandboxing of the applications.

Ubuntu Core 16 has taken full advantage of the Snaps, even the kernel and the OS itself are delivered as Snaps. The Ubuntu Core OS contains just the base file systems. Its image size is almost half of the CentOS Atomic Host 7.

The transactional nature of the Snap package updates is a boon for developers. If the update fails, an automatic rollback is initiated which encourages developers to update their applications more often and without any fear of a crash. Using Update Control, they can validate an update in the ecosystem before applying it.

The lightweight Ubuntu Core 16 also features a Snap app store that allows developers to publish apps for various internet-connected devices. The Snappy Ubuntu Core 16 is intended for single board computers, SoCs and cloud platforms. It has already been deployed in top-of-rack switches, drones, radio access networks, gateways, etc. Here is a list of supported devices :

- Raspberry Pi 2 and 3
- Qualcomm DragonBoard 410c
- Intel NUC
- Intel Joule
- Samsung Artik
- KVM

You can download the Snappy Ubuntu Core 16 ISO on Ubuntu's website.

<https://fossbytes.com/snappy-ubuntu-core-16-launched-for-iot-devices-raspberry-pi-cloud/>



## Microsoft Releases Open Source Toolkit That Understands Words Just Like Humans, Adds C++ & Python Support



### **Microsoft Releases Open Source Toolkit That Understands Words Just Like Humans, Adds C++ & Python Support.**

Microsoft flaunted their new speech recognition system that can recognize words in a conversation to the same extent as a human would do. It was able to achieve a low word error rate (WER) of 5.9 percent.

The system is built using the Microsoft Cognitive

Toolkit, previously known as CNTK toolkit, an in-house project by Microsoft researchers for their personal use. They created it for speech and image recognition. However, the advantages of the toolkit were felt and it was taken beyond image and speech recognition.

The cognitive toolkit helps researchers with neural networks leading to the creation of their machine learning systems which can run on computers with traditional CPUs and GPUs. Data sets of variety in size can be processed by the Microsoft Cognitive Toolkit on either a single machine or a series of computers in a data center. The toolkit enables improved performance on Pascal architecture-based GPUs in the Nvidia DGX-1.

The updated toolkit can be used as a library with C++ and Python APIs. It also enables reinforcement learning research for machine learning systems which involves training them to do a particular task by trial and error method. This would allow AI agents to take complex decisions.

A Germany-based company, Liebherr, has tried to use the Microsoft Cognitive Toolkit by installing cameras in their refrigerators. The cameras can detect what food is present inside and an inventory list created automatically. The prospects indicate shopping and meal planning based on the monitoring of available food items.

Microsoft Cognitive Toolkit has been used by the Bing team to understand the context of search queries. For instance, when a user types “How to make an apple pie?”, Bing should be able to understand that the search is made for the recipe even though the word “recipe” isn’t included in the search query.

Furthermore, Microsoft’s latest open source toolkit gives competition to existing developments such as Google’s TensorFlow. The Toolkit allows a system to be scalable and reduces the training time. The Microsoft Cognitive Toolkit is available on GitHub as beta.

<https://fossbytes.com/microsofts-breakthrough-open-source-cognitive-toolkit-just-got-a-major-upgrade/>



## xfce4-panel 4.12.1 Released, Xfce 4.14 Still A Long Ways Out

Xfce4-panel 4.12.1 has been released as a "long overdue maintenance release" while Xfce 4.14 is still at its infancy.

Xfce4-panel 4.12.1 has translation updates, support for xfpanel-switch in the preferences and some basic fixes. This comes a few weeks after the quiet bug-fix releases of xfce4-settings 4.12.1 and also joined by the xfconf 4.12.1 release this week.

But while it's close to two years since Xfce 4.12.0 was officially released, there is still no sign of Xfce 4.14 release, and it appears to be quite a ways still out. Xfce 4.14 is still supposed to focus on finishing the porting from GTK2 to GTK3, making use of GDBus, replacing deprecated widgets and other modernization updates.

The Xfce.org Wiki Roadmap still doesn't have any dates firmed up for Xfce 4.14 targets. The page seems to indicate as well that Xfce 4.14 won't be around the corner. But once there is something new to report on Xfce 4.14, you can certainly expect to read about it on Phoronix while still being fans of this lightweight GTK desktop. Albeit, a pity it's not yet fully living in a modern GTK3 world.

[http://www.phoronix.com/scan.php?page=news\\_item&px=Xfce4-Panel-4.12.1](http://www.phoronix.com/scan.php?page=news_item&px=Xfce4-Panel-4.12.1)

## Google Announces Code-In 2016 to Encourage Open Source

**Google Code-in is an annual online contest hosted by Google for pre-university students aged between 13 to 17 years. The participants have to pick small tasks from various open source organizations who also happen to be their mentors. The grand prize winners will be invited for a trip at the Mountain View Campus.**

The year 2016 is about to end in a couple of months and Google is back with its fall Code-in ritual. Just like the previous years, this year's Google Code-in 2016 contest will involve numerous bite-sized tasks from participating organizations that specialize in open source projects. They will also act as the mentors for the pre-university students (aged 13 to 17) taking part in the contest.

Other than being a determined internet company, Google also loves open source and is among the regular contributors to the open source community. They recently open sourced their Show and Tell A.I system which can write captions for an image and new open source font called Noto which supports 800+ languages.

The Google Code-in 2016 online contest will start on November 28, 2016 (17:00 UTC).



The participant kids will have to choose tasks (ranging between 3 to 5 hours) from one of the following categories:

- Coding
- Documentation/Training
- Outreach/Research
- Quality Assurance
- User Interface

The participant can seek help from mentor organizations whose task they've claimed and are meant to submit it before the set deadline (January 16, 2017). Google has chosen a total of 17 open source organizations which include KDE, Wikimedia, OpenMRS, Drupal, Copyleft Games, etc.

The tasks will be evaluated by the respective mentor organizations and participants with one successful submission will receive a digital certificate. The ones with three or more will receive a t-shirt. For each organization, five finalists will get a Google Code-in Hoodie.

The grand prize – for the two finalists in every organization – will earn a trip to Google's Mountain View Campus which will happen sometime in June 2017. The name of the finalists and the grand prize winners will be announced on January 30, 2017.

Visit the Google Code-in 2016 website to know more about the contest.

<https://fossbytes.com/google-code-in-2016-annocement-open-source/>

## Project Darling is Still Trying to Run MacOS/OSX Software on Linux

Lately, work on Darling seems to have picked up after a brief hiatus.

In late 2013, it looked like Darling development stalled but in January of 2014; news concerning the project had been refreshed. But that was the last time I had anything to report on the project. Until recently, hearing from a Phoronix reader and saving it for an otherwise slow news day, that Darling is in fact under development.



The Darling Git repository remains active. As of writing this article, there has been over 1,200 commits with the most recent activity being just two days ago. The most recent code additions were adding an initial GDB JIT interface, adding a Ruby sub-module and some documentation updates.

Darling is still progressing. Nonetheless, in its latest state, it cannot run any MacOS GUI applications but rather only basic command-line apps with both 32-bit and 64-bit capabilities. From the Darling Shell, there is support for working with DMG images and even using Apple's Xcode tool-chain for compiling basic "Hello World!" type applications for MacOS and running from a Linux system.

For more information on the Darling project, visit the project site at DarlingHQ.org.

[http://www.phoronix.com/scan.php?page=news\\_item&px=Darling-2016-Still-Going](http://www.phoronix.com/scan.php?page=news_item&px=Darling-2016-Still-Going)

## Thank you!

The FreeBSD Foundation is grateful for all the generous donations from individuals, organizations and businesses over the years. The Foundation is fully funded by these donations and without them, we would not exist.

IRIDIUM | \$100,000 - \$249,999





GOLD | \$25,000 - \$49,999

facebook

NETFLIX

SILVER | \$10,000 - \$24,999

*acceleration*systems

ARM

Google

iXsystems

Tarsnap

vmware

<https://www.freebsdoundation.org/donors/>



# Great Specials

On FreeBSD® & PC-BSD® Merchandise

Give us a call & ask about our  
SOFTWARE BUNDLES

1.925.240.6652

**\$39.95**

FreeBSD 9.1 Jewel Case CD Set  
or FreeBSD 9.1 DVD

**\$29.95**

PC-BSD 9.1 DVD

**\$49.95**

The PC-BSD 9.0 Users Handbook  
PC-BSD 9.1 DVD

**\$99.95**

The FreeBSD CD or DVD Bundle

Inside each CD/DVD Bundle, you'll find:  
FreeBSD Handbook, 3rd Edition  
Users Guide FreeBSD Handbook, 3rd Edition, Admin Guide  
FreeBSD 9.1 CD or DVD set  
FreeBSD Toolkit DVD



*Stylish Dress Attire*  
Look Your Professional Best



*Comfy Apparel*  
Stay Warm in Zip Ups & Pullovers

*T-Shirts*  
Lots of Styles to Choose From

**FreeBSD 9.1 Jewel Case CD/DVD ..... \$39.95**

CD Set Contains:

- Disc 1** Installation Boot LiveCD (i386)
- Disc 2** Essential Packages Xorg (i386)
- Disc 3** Essential Packages, GNOME2 (i386)
- Disc 4** Essential Packages (i386)

FreeBSD 9.0 CD ..... \$39.95

FreeBSD 9.0 DVD ..... \$39.95

## FreeBSD Subscriptions

Save time and \$\$\$ by subscribing to regular updates of FreeBSD

FreeBSD Subscription, start with CD 9.1 ..... \$29.95

FreeBSD Subscription, start with DVD 9.1 ..... \$29.95

FreeBSD Subscription, start with CD 9.0 ..... \$29.95

FreeBSD Subscription, start with DVD 9.0 ..... \$29.95

## PC-BSD 9.1 DVD (Isotope Edition)

PC-BSD 9.1 DVD ..... \$29.95

PC-BSD Subscription ..... \$19.95

## The FreeBSD Handbook

The FreeBSD Handbook, Volume 1 (User Guide) ..... \$39.95

The FreeBSD Handbook, Volume 2 (Admin Guide) ..... \$39.95

## The FreeBSD Handbook Specials

The FreeBSD Handbook, Volume 2 (Both Volumes) ..... \$59.95

The FreeBSD Handbook, Both Volumes & FreeBSD 9.1 ..... \$79.95

**PC-BSD 9.0 Users Handbook ..... \$24.95**

**BSD Magazine ..... \$11.99**

**The FreeBSD Toolkit DVD ..... \$39.95**

**FreeBSD Mousepad ..... \$10.00**

**FreeBSD & PCBSD Caps ..... \$20.00**

**BSD Daemon Horns ..... \$2.00**



*Bundle Specials!*  
Save \$\$\$

*Just Plain Fun*  
Mousepads & Novelty Horns



*BSD Magazine*  
Available Monthly



For even MORE items  
visit our website today!

[www.FreeBSDMall.com](http://www.FreeBSDMall.com)



## First Look at the Renewed CTL High Availability Implementation in FreeBSD

*by Mikhail Zakharov*

**This enhancement looks extremely important for the BeaST storage system, as implementation of high available native ALUA in FreeBSD can potentially replace the BeaST arbitration mechanism (“Arbitrator”), which is completely described in the papers on the BeaST project page.**

So let’s see what has happened. According to the `ctl(4)` man page: “The `ctl` subsystem provides SCSI disk and processor emulation” and “serves as a kernel component of the native iSCSI target”. Among other features, it has now reimplemented “High Availability clustering support with ALUA”. See source code revision 287621 by Alexander Motin for more details on the change. Actually, this revision was done a year ago and the feature is available both in FreeBSD 11.0 and in 10.3 releases.

ALUA in storage world terminology means Asymmetric Logical Unit Assignment. In simple words, this set of technologies allows a host to access any LUN via both controllers of a storage system.

FreeBSD `ctl(4)` man-page claims all possible modes are now available:

```
"kern.cam.ctl.ha_mode
```

```
Specifies High Availability cluster operation mode:
```

```
0    Active/Standby -- primary node has backend access and
```

```
processes requests, while secondary can only
```



```
do basic
    LUN discovery and reservation;

    1 Active/Active -- both nodes have backend ac-
cess and
    process requests, while secondary node synchro-
nizes
    processing with primary one;

    2 Active/Active -- primary node has backend ac-
cess and
    processes requests, while secondary node for-
wards all
    requests and data to primary one;"
```

Now let's see if it is reasonable to use this new HA storage functionality in the BeaST project.

As I still do not have any real hardware drive-enclosures, we will use Oracle Virtual Box and iSCSI protocol. I have already deployed this environment for the BeaST development, so we can use the similar, yet more simplified, template for the renewed CTL HA testing purpose.

We will run two storage controllers (ctrl-a, ctrl-b) and a host (cln-1). A virtual SAS drive (da0) of 256 MB is configured as "shareable" in Virtual Media Manager and simultaneously connected with both storage controllers.

Two virtual LANs are configured:

- 192.168.10.0/24 is a private network for HA interconnect;
- 192.168.20.0/24 is for the public access to the front-end and LUNs.

These IP addresses are assigned to the hosts:

Host	Private network	Public network
ctrl-a	192.168.10.101	192.168.20.101
ctrl-b	192.168.10.102	192.168.20.102
cln-1	-	192.168.20.103

We will use the fresh FreeBSD 11.0 release:

```
% uname -a

FreeBSD ctrl-a 11.0-RELEASE-p1 FreeBSD 11.0-RELEASE-p1 #0 r306420:
Thu Sep 29 01:43:23 UTC 2016
    root@releng2.nyi.freebsd.org:/usr/obj/usr/src/sys/GENERIC amd64
```

Before doing anything else, let's save CTL frontend ports states of both controllers to refer to them later during our experiments:

```
root@ctrl-a:/home/beast # ctladm portlist

Port Online Frontend Name      pp vp
0     YES   ioctl    ioctl    0  0
1     YES   tpc      tpc      0  0
2     NO    camsim   camsim   0  0  naa.50000006e879fb03

root@ctrl-b:/home/ beast # ctladm portlist

Port Online Frontend Name      pp vp
128  YES   ioctl    ioctl    0  0
129  YES   tpc      tpc      0  0
130  NO    camsim   camsim   0  0  naa.50000001f740eb83
```



Now we can start configuring controllers. First of all, add essential variables to the `/boot/loader.conf`, then reboot both controllers:

ctrl-a	ctrl-b
<pre>ctl_load="YES"  kern.cam.ctl.ha_id=1  kern.cam.ctl.ha_mode=1  kern.cam.ctl.ha_role=0  kern.cam.ctl.iscsi.ping_timeout=0</pre>	<pre>ctl_load="YES"  kern.cam.ctl.ha_id=2  kern.cam.ctl.ha_mode=1  kern.cam.ctl.ha_role=1  kern.cam.ctl.iscsi.ping_timeout=0</pre>

Where:

`ctl_load="YES"` loads the CTL driver as module.

`kern.cam.ctl.ha_id` – specifies the node ID (1 or 2) and 0 disables HA functionality.

`kern.cam.ctl.ha_mode` – sets operational mode. See the description of it at the beginning of the article. For our purposes, we are interested in Active/Active modes only.

`kern.cam.ctl.ha_role` – configures default role for the node. So ctrl-a is set as 0 (primary node), ctrl-b – 1 (secondary node). The role also can be specified on per-LUN basis which allows to distribute LUNs over both controllers evenly.

`kern.cam.ctl.iscsi.ping_timeout` – is a number of seconds to wait for initiator’s response on NOP-In PDU, which is issued by the target when the traffic is inactive. By default, the session is forcibly terminated after 5 second. But for testing purposes we disable it by specifying 0.

Note, `kern.cam.ctl.ha_id` and `kern.cam.ctl.ha_mode` are read-only parameters and must be set only via the `/boot/loader.conf` file. Other useful variables we can put in `/etc/sysctl.conf` but I choose only `kern.cam.ctl.debug=1`:

```
beast@ctrl-a:~ % grep kern.cam.ctl /etc/sysctl.conf
```

```
kern.camctl.debug=1

#kern.camctl.ha_peer="connect 192.168.10.102:7777"

beast@ctrl-b:~ % grep kern.camctl /etc/sysctl.conf

kern.camctl.debug=1

#kern.camctl.ha_peer="listen 192.168.10.102:7777"
```

As you can see, there is a remark sign before kern.camctl.ha\_peer variable. It is done to prevent an attempt to start CTL HA connection at boot: at this point LAN interfaces are not up yet, so the connection will fail.

But we can lately start CTL HA interconnect manually from shell or by script:

ctrl-a	ctrl-b
# sysctl kern.camctl.ha_peer="connect 192.168.10.102:7777"	sysctl kern.camctl.ha_peer="listen 192.168.10.102:7777"

If everything is OK, we will see these messages in logs on both controllers:

```
root@ctrl-b:/home/beast # dmesg | tail -2

CTL: HA link status changed from 0 to 1

CTL: HA link status changed from 1 to 2
```

The link states can be: 0 – not configured, 1 – configured but not established and 2 – established. The link state information can be also checked via sysctl:

```
root@ctrl-a:/home/beast # sysctl kern.camctl.ha_link

kern.camctl.ha_link: 2
```



As we got link state 2, we can check what is going on the CTL frontend:

```
root@ctrl-a:/home/beast # ctladm portlist

Port Online Frontend Name      pp vp
0     YES  ioctl  ioctl      0  0
1     YES  tpc    tpc        0  0
2     NO   camsim camsim      0  0  naa.50000006e879fb03
128   YES  ha     2:ioctl    0  0
129   YES  ha     2:tpc      0  0
130   NO   ha     2:camsim   0  0  naa.50000001f740eb83

root@ctrl-b:/home/beast # ctladm portlist

Port Online Frontend Name      pp vp
0     YES  ha     1:ioctl    0  0
1     YES  ha     1:tpc      0  0
2     NO   ha     1:camsim   0  0  naa.50000006e879fb03
128   YES  ioctl  ioctl      0  0
129   YES  tpc    tpc        0  0
130   NO   camsim camsim      0  0  naa.50000001f740eb83
```

As you can see, HA interconnection is established. It means we can add some LUNs to use them on our client host. Therefore, create simple `/etc/ctl.conf` to add appropriate definitions for our iSCSI targets:

ctrl-a	ctrl-b
portal-group pg0 {  discovery-auth-group no-authentication	portal-group pg0 {  discovery-auth-group no-authentication

ctrl-a	ctrl-b
<pre>listen 192.168.20.101  }  target iqn. 2016-01.local.beast:target0 {      auth-group no- authentication      portal-group pg0      lun 1 {          path /dev/da0      }  }</pre>	<pre>listen 192.168.20.102  }  target iqn. 2016-01.local.beast:target0 {      auth-group no- authentication      portal-group pg0      lun 1 {          path /dev/da0      }  }</pre>

Then start ctld on both controllers:

```
# service ctld onestart
```

Now check what is registered on ports:

```
root@ctrl-a:/home/beast # ctladm portlist

Port Online Frontend Name      pp vp
0     YES   ioctl   ioctl   0  0
1     YES   tpc     tpc     0  0
2     NO    camsim  camsim  0  0  naa.5000000d5de41b03
3     YES   iscsi   iscsi   257 1
      iqn.2016-01.local.beast:target0,t,0x0101
128  YES    ha      2:ioctl 0  0
```



```
129  YES    ha      2:tpc    0  0

130  NO     ha      2:camsim 0  0  naa.500000091b30b383

131  YES    ha      2:iscsi  257 1
iqn.2016-01.local.beast:target0,t,0x0101

root@ctrl-b:/home/beast # ctladm portlist

Port Online Frontend Name      pp vp
0     YES    ha      1:iocctl 0  0
1     YES    ha      1:tpc    0  0
2     NO     ha      1:camsim 0  0  naa.5000000d5de41b03
3     YES    ha      1:iscsi  257 1
iqn.2016-01.local.beast:target0,t,0x0101

128  YES    iocctl  iocctl  0  0
129  YES    tpc     tpc     0  0
130  NO     camsim  camsim  0  0  naa.500000091b30b383
131  YES    iscsi   iscsi   257 1
iqn.2016-01.local.beast:target0,t,0x0101
```

As new LUNs are shown, everything is going well right now. So we can start the client host and establish iSCSI connection with our storage:

```
root@cln-1:/home/beast # service iscsid onestart

root@cln-1:/home/beast # sysctl kern.iscsi.fail_on_disconnection=1

root@cln-1:/home/beast # iscsictl -A -p 192.168.20.101 -t
iqn.2016-01.local.beast:target0

root@cln-1:/home/beast # iscsictl -A -p 192.168.20.102 -t
iqn.2016-01.local.beast:target0
```

Note, `sysctl kern.iscsi.fail_on_disconnection=1` on the client is needed to drop connection with one of the controllers in case of its failure.

During this operation, we can see log updates with activity on iSCSI LUNs:

```
On ctrl-a:
```

```
(0:3:1/0): MODE SENSE(6). CDB: 1a 00 0a 00 18 00  Tag: 0x4/0
(0:3:1/0): CTL Status: SCSI Error
(0:3:1/0): SCSI Status: Check Condition
(0:3:1/0): SCSI sense: UNIT ATTENTION asc:29,1 (Power on occurred)
```

```
On ctrl-b:
```

```
(0:131:1/0): MODE SENSE(6). CDB: 1a 00 0a 00 18 00  Tag: 0x4/0
(0:131:1/0): CTL Status: SCSI Error
(0:131:1/0): SCSI Status: Check Condition
(0:131:1/0): SCSI sense: UNIT ATTENTION asc:29,1 (Power on occurred)
```

```
And on cln-1:
```

```
da0 at iscsi1 bus 0 scbus3 target 0 lun 1
da0: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device
da0: Serial Number MYSERIAL    0
da0: 150.000MB/s transfers
da0: Command Queueing enabled
da0: 256MB (524288 512 byte sectors)
da1 at iscsi2 bus 0 scbus4 target 0 lun 1
```



```
da1: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device
da1: Serial Number MYSERIAL    0
da1: 150.000MB/s transfers
da1: Command Queueing enabled
da1: 256MB (524288 512 byte sectors)
```

So we can state that the client has reached both LUNs (actually the client has accessed the same physical drive through connections with two different controllers).

As we know that da0 and da1 on the client are the same drive, we can put them under multipathing control:

```
root@cln-1:/home/beast # gmultipath create -A HA /dev/da0 /dev/da1
```

Note, option -A enables Active/Active mode of multipathing, so the workload will be distributed over both paths and controllers.

Check if we succeeded:

```
root@cln-1:/home/beast # gmultipath status
```

Name	Status	Components
multipath/HA	OPTIMAL	da0 (ACTIVE)
		da1 (ACTIVE)

Well, now let's create a new filesystem and mount it:

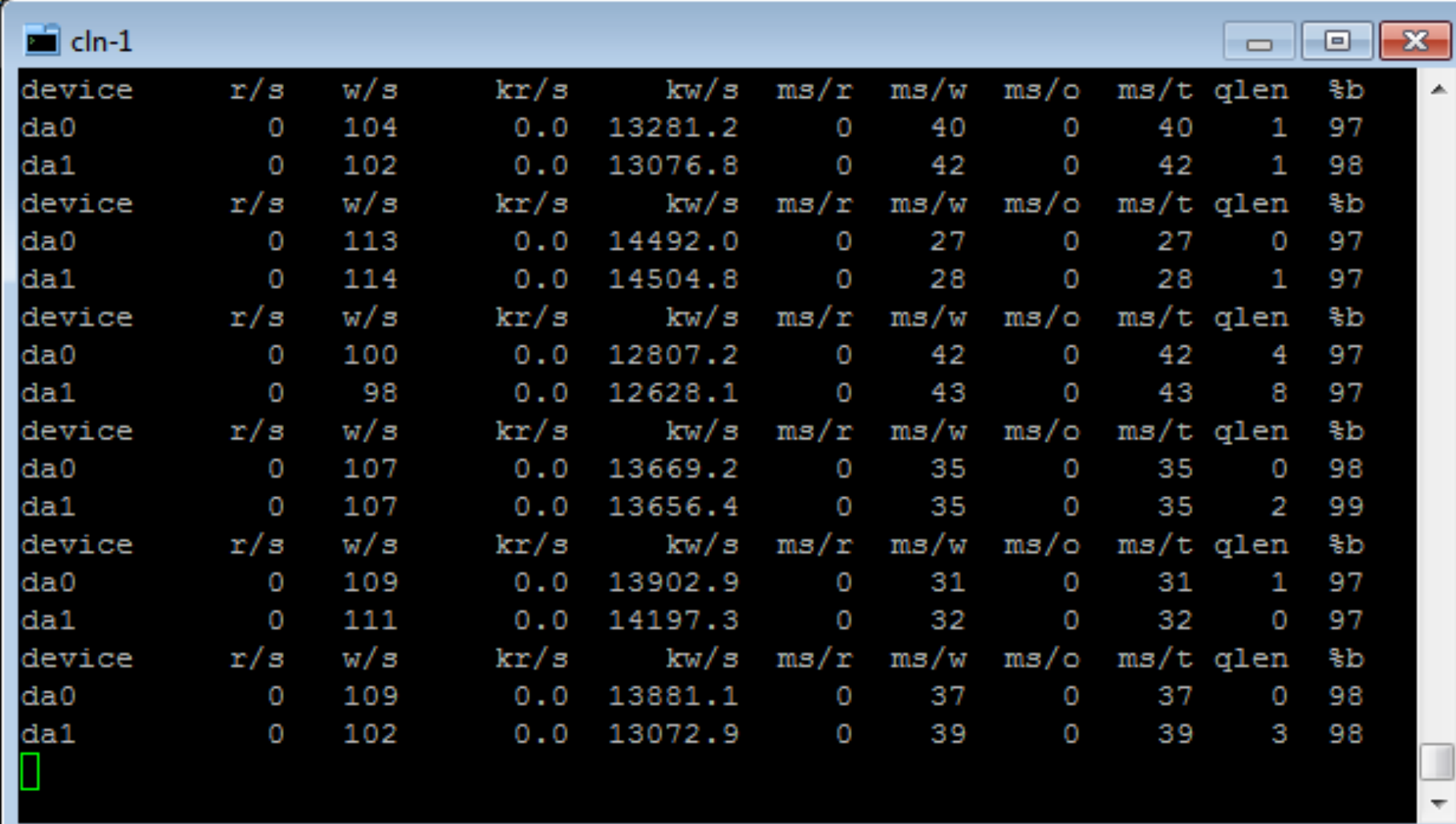
```
root@cln-1:/home/beast # newfs /dev/multipath/HA
root@cln-1:/home/beast # mount /dev/multipath/HA /mnt
```

Now we can force the construction to work, so let's continuously copy a file:

```
# while true; do cp ports.tar.gz /mnt; done
```

And check the results from the client side:

```
root@cln-1:/home/beast # iostat -xd 5 | grep '^d'
```



device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	104	0.0	13281.2	0	40	0	40	1	97
da1	0	102	0.0	13076.8	0	42	0	42	1	98
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	113	0.0	14492.0	0	27	0	27	0	97
da1	0	114	0.0	14504.8	0	28	0	28	1	97
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	100	0.0	12807.2	0	42	0	42	4	97
da1	0	98	0.0	12628.1	0	43	0	43	8	97
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	107	0.0	13669.2	0	35	0	35	0	98
da1	0	107	0.0	13656.4	0	35	0	35	2	99
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	109	0.0	13902.9	0	31	0	31	1	97
da1	0	111	0.0	14197.3	0	32	0	32	0	97
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	109	0.0	13881.1	0	37	0	37	0	98
da1	0	102	0.0	13072.9	0	39	0	39	3	98

According to the Active/Active multipathing mode, both devices are working.

Now let's check what is going on storage controllers (ctrl-a and ctrl-b):

```
# iostat -xd 5 | egrep '^dev|^da0|^cb'
```



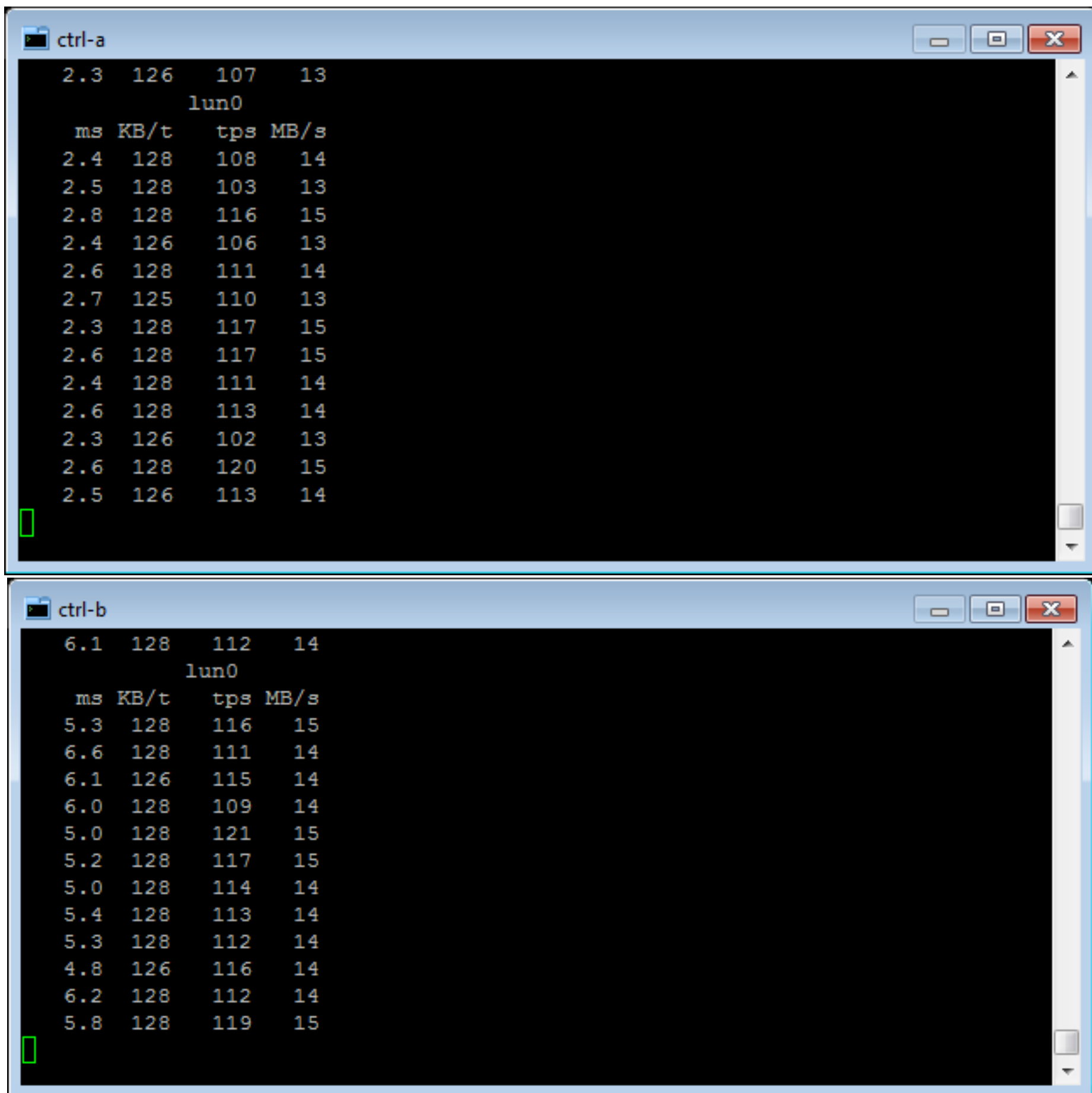
As we have set full Active/Active CTL HA mode (`kern.camctl.ha_mode=1`) for the cluster, we can see the similar picture on both controllers:

ctrl-a											
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	113	0.0	14481.1	0	3	0	3	0	29	
cbb0	0	113	0.0	14481.1	0	3	0	3	0	30	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	115	0.0	14631.4	0	2	0	2	1	25	
cbb0	0	115	0.0	14631.4	0	2	0	2	1	26	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	112	0.0	14401.7	0	2	0	2	0	27	
cbb0	0	112	0.0	14401.7	0	2	0	2	0	28	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	91	0.0	11647.5	0	22	0	22	1	50	
cbb0	0	91	0.0	11647.5	0	22	0	22	1	50	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	96	0.0	12309.9	0	19	0	19	1	46	
cbb0	0	96	0.0	12309.9	0	19	0	19	1	46	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	109	0.0	13915.9	0	12	0	12	2	42	
cbb0	0	109	0.0	13915.9	0	12	0	12	2	42	

ctrl-b											
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	103	0.0	13295.3	0	11	0	11	1	34	
cbb0	0	103	0.0	13295.3	0	11	0	11	1	35	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	111	0.0	14259.3	0	2	0	2	1	23	
cbb0	0	111	0.0	14259.3	0	2	0	2	1	23	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	94	0.0	11960.9	0	14	0	14	7	38	
cbb0	0	94	0.0	11960.9	0	14	0	14	7	38	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	104	0.0	13298.9	0	8	0	8	1	30	
cbb0	0	104	0.0	13298.9	0	8	0	8	1	31	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	106	0.0	13510.3	0	9	0	9	0	33	
cbb0	0	106	0.0	13510.3	0	9	0	9	0	34	
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b	
da0	0	103	0.0	13133.1	0	12	0	12	0	37	
cbb0	0	103	0.0	13133.1	0	12	0	12	0	37	

Now let's see CTL LUN statistics on both controllers. So run:

```
# ctlstat -C
```



The image shows two terminal windows side-by-side, each displaying the output of the `ctlstat -C` command for a specific controller. The top window is titled 'ctrl-a' and the bottom window is titled 'ctrl-b'. Both windows show a table of statistics for LUN0, including latency (ms), transfer size (KB/t), throughput (tps), and transfer rate (MB/s). The data for ctrl-a shows much lower latency (around 2.3-2.8 ms) compared to ctrl-b (around 4.8-6.6 ms), illustrating a performance difference between the two nodes.

ctrl-a			
lun0			
ms	KB/t	tps	MB/s
2.3	126	107	13
2.4	128	108	14
2.5	128	103	13
2.8	128	116	15
2.4	126	106	13
2.6	128	111	14
2.7	125	110	13
2.3	128	117	15
2.6	128	117	15
2.4	128	111	14
2.6	128	113	14
2.3	126	102	13
2.6	128	120	15
2.5	126	113	14

ctrl-b			
lun0			
ms	KB/t	tps	MB/s
6.1	128	112	14
5.3	128	116	15
6.6	128	111	14
6.1	126	115	14
6.0	128	109	14
5.0	128	121	15
5.2	128	117	15
5.0	128	114	14
5.4	128	113	14
5.3	128	112	14
4.8	126	116	14
6.2	128	112	14
5.8	128	119	15

And it seems, the second node for the LUN responds slower (two times slower in our example).



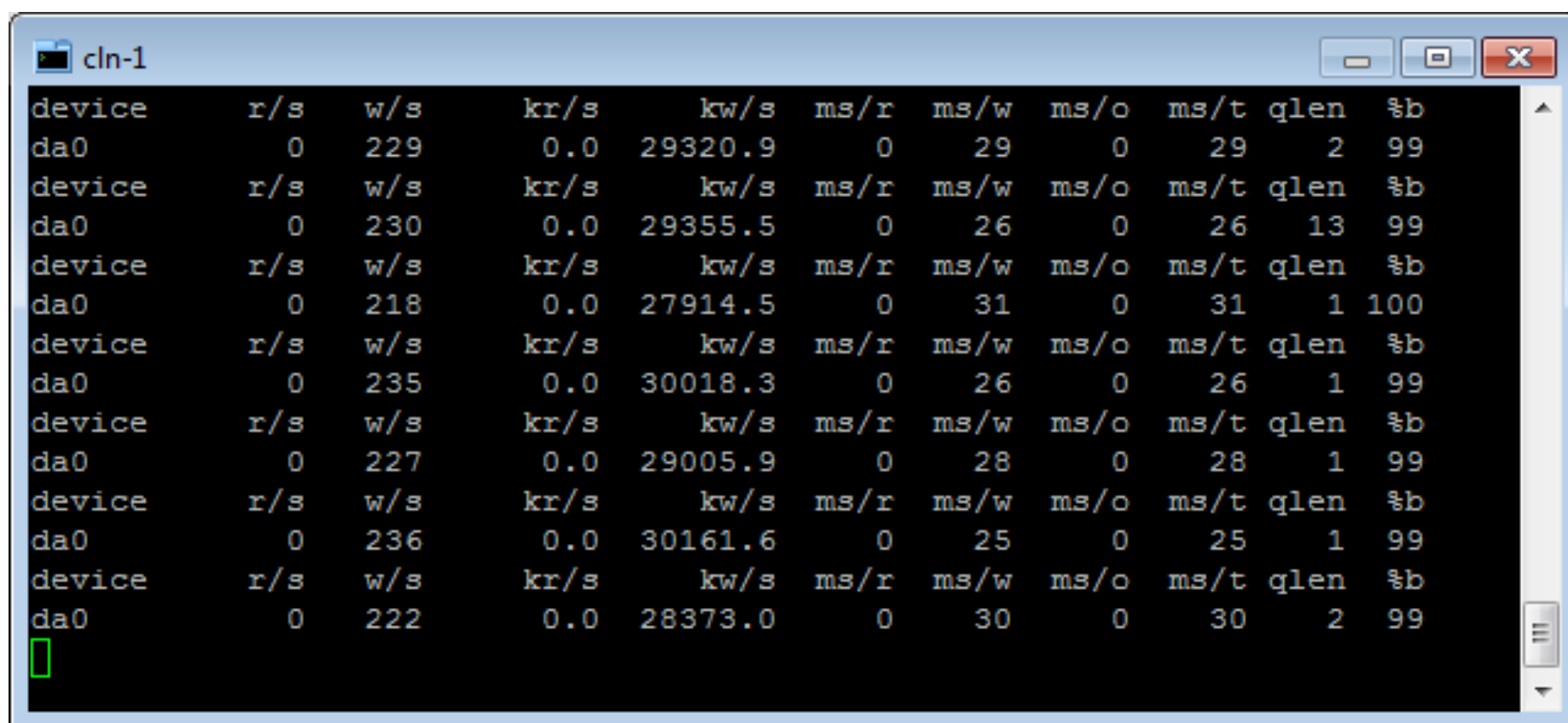
Remember that `kern.camctl.ha_mode=2` enables Active/Active frontend, while the secondary node forwards all requests and data to the primary one. I do not show screenshots, so you have to believe me, but if we set this mode, the overall picture will be similar except that `iostat -xd 5` on the secondary node shows zero disk activity and the primary node processes all the workload.

Anyway, the full Active/Active CTL HA configuration (`kern.camctl.ha_mode=1`) runs well, utilizing all available paths through both controllers. Now it's time to test high availability of the new CTL HA subsystem.

I have no idea of how to fail backend virtual link between the drive and SAS controller in Virtual-Box environment, but we can easily simulate a crash of a whole controller. So, let's just shutdown a secondary node (`ctrl-b`) and see if the cluster can survive it:

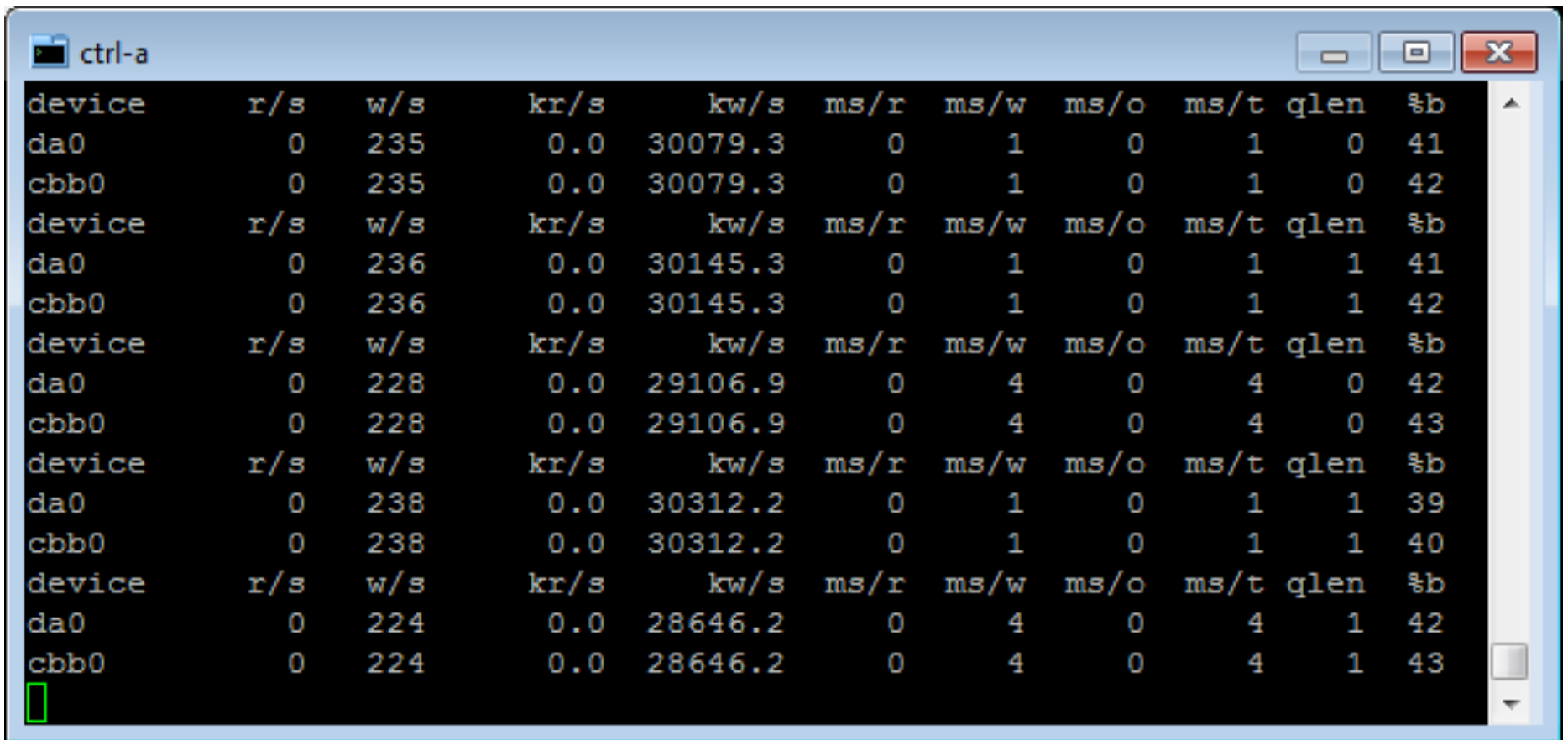
```
root@ctrl-b:/home/beast # shutdown -p now
```

And nothing bad has happened to the client. It has lost one iSCSI path (for the `da1`) but multipathing works well, it is now forwarding all the data through the primary node (`ctrl-a`):



device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	229	0.0	29320.9	0	29	0	29	2	99
da0	0	230	0.0	29355.5	0	26	0	26	13	99
da0	0	218	0.0	27914.5	0	31	0	31	1	100
da0	0	235	0.0	30018.3	0	26	0	26	1	99
da0	0	227	0.0	29005.9	0	28	0	28	1	99
da0	0	236	0.0	30161.6	0	25	0	25	1	99
da0	0	222	0.0	28373.0	0	30	0	30	2	99

And on ctrl-a (primary node of CTL HA cluster), we can see that data is going to the da0:



device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	235	0.0	30079.3	0	1	0	1	0	41
cbb0	0	235	0.0	30079.3	0	1	0	1	0	42
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	236	0.0	30145.3	0	1	0	1	1	41
cbb0	0	236	0.0	30145.3	0	1	0	1	1	42
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	228	0.0	29106.9	0	4	0	4	0	42
cbb0	0	228	0.0	29106.9	0	4	0	4	0	43
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	238	0.0	30312.2	0	1	0	1	1	39
cbb0	0	238	0.0	30312.2	0	1	0	1	1	40
device	r/s	w/s	kr/s	kw/s	ms/r	ms/w	ms/o	ms/t	qlen	%b
da0	0	224	0.0	28646.2	0	4	0	4	1	42
cbb0	0	224	0.0	28646.2	0	4	0	4	1	43

Our next steps will be to boot the ctrl-b controller, restore our CTL HA cluster and start copying `ports.tar.gz` file once again. But I'm not going to show it here to save time and space.

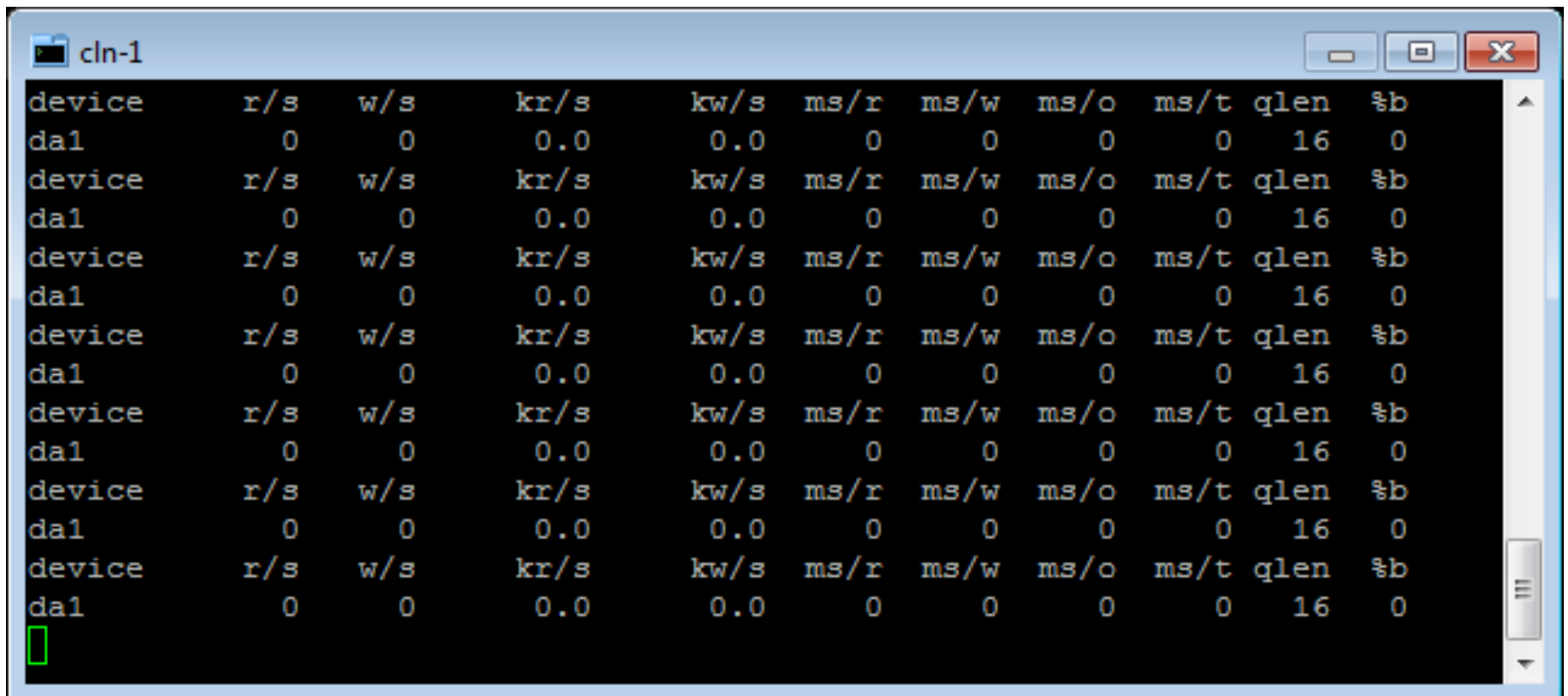
After restoring the cluster and starting `iostat -xd 5` along with `ctlstat -C`, we can finally crash the primary node (ctrl-a):

```
root@ctrl-a:/home/beast # shutdown -p now
```

And things turn really bad now.

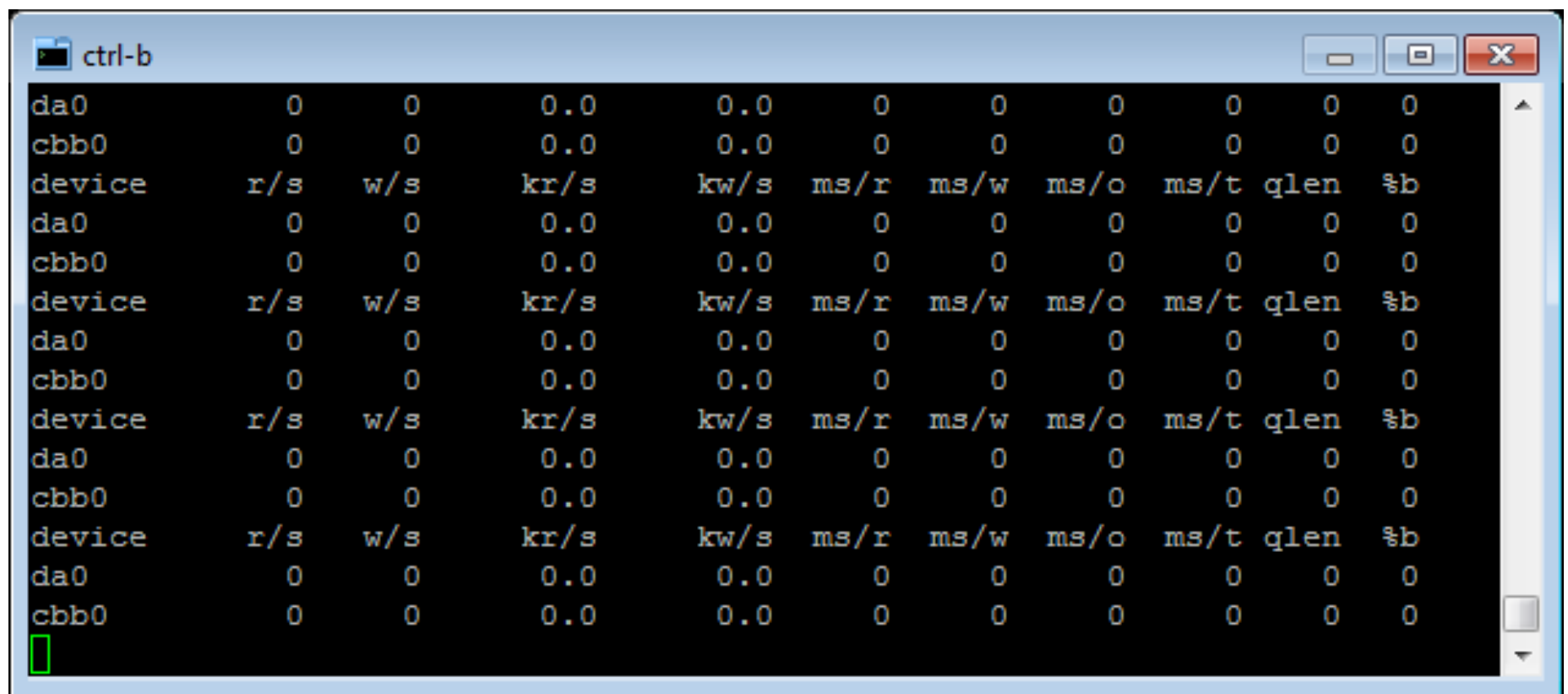


First of all, we have an active path, but traffic has absolutely stopped on the client (cln-1):



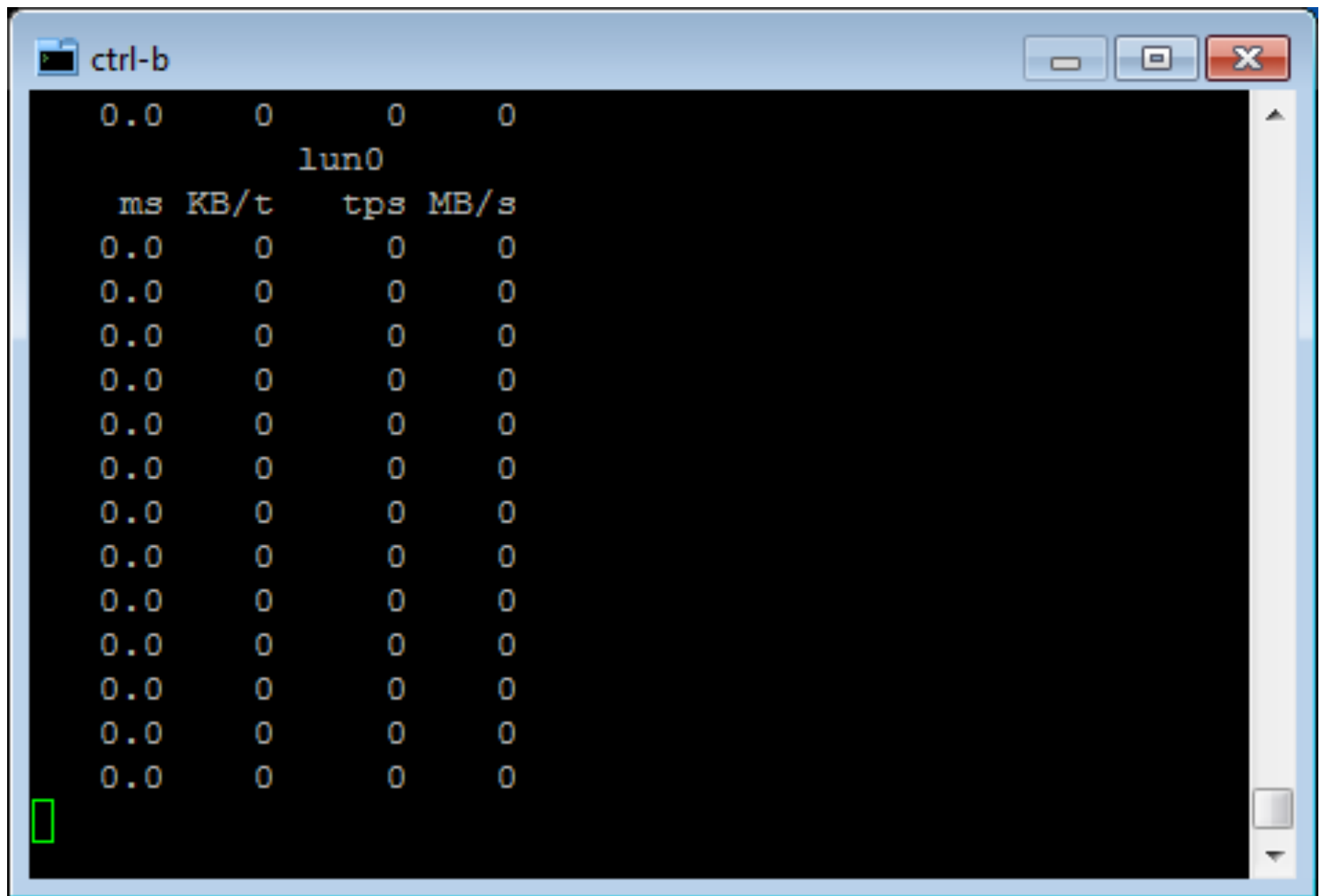
```
cln-1
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da1          0      0      0.0      0.0      0        0        0        0      16      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da1          0      0      0.0      0.0      0        0        0        0      16      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da1          0      0      0.0      0.0      0        0        0        0      16      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da1          0      0      0.0      0.0      0        0        0        0      16      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da1          0      0      0.0      0.0      0        0        0        0      16      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da1          0      0      0.0      0.0      0        0        0        0      16      0
```

Output of iostat -xd5 on ctrl-b is also empty:



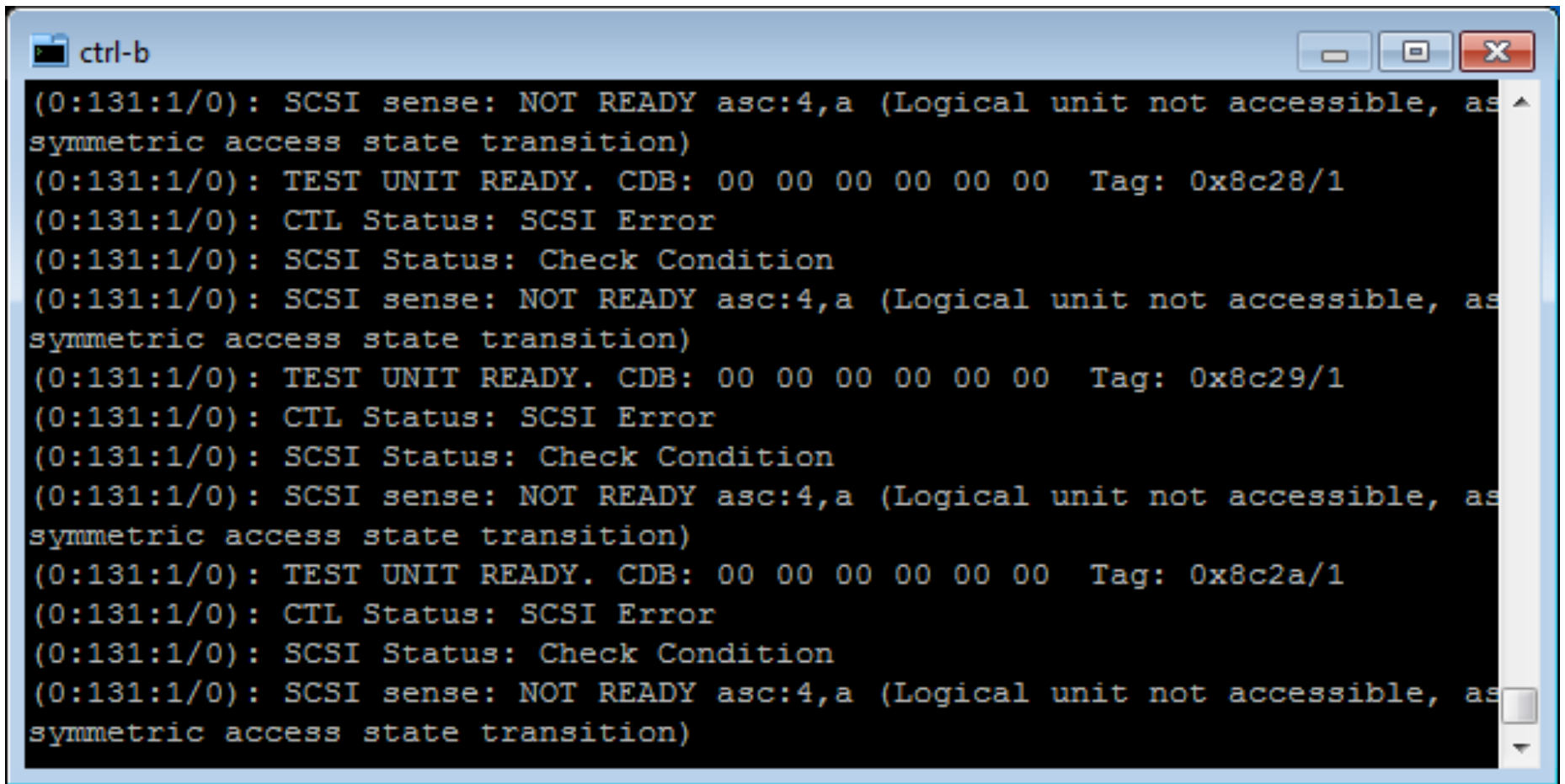
```
ctrl-b
da0          0      0      0.0      0.0      0        0        0        0      0      0
cbb0         0      0      0.0      0.0      0        0        0        0      0      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da0          0      0      0.0      0.0      0        0        0        0      0      0
cbb0         0      0      0.0      0.0      0        0        0        0      0      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da0          0      0      0.0      0.0      0        0        0        0      0      0
cbb0         0      0      0.0      0.0      0        0        0        0      0      0
device      r/s    w/s    kr/s    kw/s    ms/r    ms/w    ms/o    ms/t    qlen    %b
da0          0      0      0.0      0.0      0        0        0        0      0      0
cbb0         0      0      0.0      0.0      0        0        0        0      0      0
```

And the same picture of the LUN statistics on ctrl-b:



lun0			
ms	KB/t	tps	MB/s
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0
0.0	0	0	0

And finally, ctrl-b log is overwhelmed by this kind of message:



```
ctrl-b
(0:131:1/0): SCSI sense: NOT READY asc:4,a (Logical unit not accessible, asymmetric access state transition)
(0:131:1/0): TEST UNIT READY. CDB: 00 00 00 00 00 00 Tag: 0x8c28/1
(0:131:1/0): CTL Status: SCSI Error
(0:131:1/0): SCSI Status: Check Condition
(0:131:1/0): SCSI sense: NOT READY asc:4,a (Logical unit not accessible, asymmetric access state transition)
(0:131:1/0): TEST UNIT READY. CDB: 00 00 00 00 00 00 Tag: 0x8c29/1
(0:131:1/0): CTL Status: SCSI Error
(0:131:1/0): SCSI Status: Check Condition
(0:131:1/0): SCSI sense: NOT READY asc:4,a (Logical unit not accessible, asymmetric access state transition)
(0:131:1/0): TEST UNIT READY. CDB: 00 00 00 00 00 00 Tag: 0x8c2a/1
(0:131:1/0): CTL Status: SCSI Error
(0:131:1/0): SCSI Status: Check Condition
(0:131:1/0): SCSI sense: NOT READY asc:4,a (Logical unit not accessible, asymmetric access state transition)
```

Note, that this message “Logical unit not accessible, asymmetric access state transition” is described in the `ctl(4)` man page:

***“If there is no primary node (both nodes are secondary, or secondary node has no connection to primary one), secondary node(s) report Transitioning state.”***

Therefore, it looks like a “normal” (`kern.camctl.ha_mode=2` shows the same results) behavior of CTL HA cluster in a case of disaster and loss of the primary node. It also:

means that a very lucky administrator can restore the failed primary controller before timeouts are elapsed.

Talking seriously, the failover can be done by setting primary role to the survived controller. So:

```
root@ctrl-b:/home/beast # sysctl kern.camctl.ha_role=0
```



And the traffic starts flowing once again:

```
ctrl-b
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  240   0.0 30612.7    0   28    0   28    1  99
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  236   0.0 30116.6    0   28    0   28    1  99
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  224   0.0 28718.0    0   34    0   34    2 100
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  236   0.0 30175.7    0   32    0   32    3  99
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  240   0.0 30576.7    0   27    0   27    2  99
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  236   0.0 30162.2    0   31    0   31    1  99
device  r/s  w/s   kr/s   kw/s  ms/r  ms/w  ms/o  ms/t  qlen  %b
da1      0  242   0.0 30892.5    0   27    0   27    3  99
█
```



## About the Author:

My name is Mikhail E. Zakharov and I am a proud SAN/storage IBMer. 10 years of experience in large SAN and storage environments: mainly Hitachi, HP and Brocade. Empty – expect-like tool author. FreeBSD enthusiast.

## Create Your First FreeBSD Kernel Module

*by Abdelhadi Khiati*

**FreeBSD is one of the biggest OSs in history. It is Unix flavored and based on the Berkeley Software Distribution. Even though Linux is dominating most of the servers market nowadays, FreeBSD still has its fair share (Netflix).**

We won't dive into the reasons why you would want to use FreeBSD as your OS for production (not the purpose of this post).

I have been lucky enough to participate in Google Summer of Code with the FreeBSD foundation. I was amazed by the community surrounding it, which was noob friendly and very helpful (thank you, FreeBSD <3).

I wanted to make a starting tutorial for people to write a simple module for kernel before diving inside more complicated kernel shizzle.

The kernel module that we will be working on is a simple event handler for the kernel. It will be composed of two parts, the event handling function and the module declaration.

The module event handler is a function that handles different events for the module, like the module being loaded, unloaded or on system shutdown.

You can find the different kind of events that the module can handle:

<https://github.com/freebsd/freebsd/blob/ac8551c9b0279945334f6cae1f7a3263d1675a3e/sys/sys/module.h#L43>

**MOD\_LOAD** is set when the module is being loaded inside the kernel.

**MOD\_UNLOAD** is set when the module is unloaded from the kernel.

**MOD\_SHUTDOWN** is set on system shutdown.

**MOD\_QUIESCE** is set when the module is about to be unloaded.

The difference between `MOD_QUIESCE` and `MOD_UNLOAD` is that the module should fail `MOD_QUIESCE` if it is currently in use, whereas `MOD_UNLOAD` should only fail if it is impossible to unload the module, for instance, because there are memory references to the module which cannot be revoked.

```
static int hellomodevent(modulet mod _unused, int event, void* arg
_unused) {

    int error = 0;

    switch(event) {

        case MODLOAD:

            uprintf("Hello World.\n");

            break;

        case MODUNLOAD:

            uprintf("Goodbye.\n");

            break;

        default:

            error = EOPNOTSUPP;

            break;

    }

    return error;

}
```

Now that we have the events handling function ready. We need to declare the `moduledata_t` to be able to use it inside `DECLARE_MODULE` macro and load it into the kernel.



It has the module name and a pointer to the event handling function.

```
static moduledata_t mod_data= {  
    "hello",  
    hello_modevent,  
    NULL  
};
```

Lastly, we need to declare the module using the `DECLARE_MODULE` macro. Which has the following structure:

```
DECLARE MODULE (name, moduledata t data, sub, order);
```

And for that we need:

**name:** The module name that will be used in the SYSINIT() call to identify the module.

**data:** The moduledata\_t structure that we already presented.

**sub:** Since we are using a driver here, so the value will be `SI_SUB_DRIVERS`, this argument will specify the type of system startup interface.

**order:** Represents the order of initialization within the subsystem, we will use the `SI_ORDER_MIDDLE` value here:

```
int error = 0;

switch(event) {

    case MOD_LOAD:

        uprintf("Hello World.\n");

        break;

    case MOD_UNLOAD:

        uprintf("Goodbye.\n");

        break;

    default:

        error = EOPNOTSUPP;

        break;

}

return error;

}

static moduledata_t hello_mod = {

    "hello",

    hello_modevent,

    NULL

};

DECLARE_MODULE(hello, hello_mod, SI_SUB_DRIVERS, SI_ORDER_MIDDLE);
```

View rawmodule.c hosted with ❤ by GitHub (Github.com).

To compile the previous file, you need to use a Makefile as following:

```
KMOD=hello

SRCS=module.c

.include<bsd.kmod.mk>

int error = 0;
```

Vew rawMakefile hosted with ❤ by GitHub

Stay tuned for more in-depth tutorials about FreeBSD kernel code.



#### About the Author:

Abdelhadi Khiati 22 years old, Master student in robotics and AI graduating in 2017, participated in Google Summer of Code 2016 with FreeBSD. Passionate about everything low-level and a performance nerd.

#### Source of the article:

<http://meltmes.kiloreux.me/create-your-first-freebsd-kernel-module/>



## Securing the Future of the Web

*by Brian Spector, CEO at MIRACL*

**Security on the Internet is long overdue for change. Whether it's by script kiddies or state actors, the Internet has shown that it can be attacked in a myriad of new and inventive ways. This poses huge risks to the future of our digital economy: nearly five billion private data records have been exposed globally since 2013, and barely a week goes by without a new data breach or vulnerability being revealed.**

Most of the problem stems from the use of outdated Internet infrastructure, like Public Key Infrastructure (PKI), which creates single points of compromise and simply cannot scale for the world we live in now. This is a problem that can't be patched – the only thing to do is start over, with a new security framework for the Internet.

The Internet has changed enormously since it was first established, and the ways that we originally tried to secure it simply don't work anymore. Take digital certificates, for example. While certificates once successfully authenticated servers, back when everyone was using a single device, they don't scale for the world we live in now. Certificates can't authenticate users across apps and mobiles, they don't work with virtualized computing, and they certainly won't scale for the Internet of Things. With 25 billion new devices set to hit the Internet by 2025, the need for a better worldwide cryptosystem for securing information is paramount. So the only question remaining is: what does the future of security on the web look like?

In May of this year, together with NTT Innovation Institute and NTT Labs, we contributed authentication code to a new open-source project within the Apache Incubator called Apache Milagro (incubating). The project seeks to provide an alternative to centralized certificates and passwords in a world that has shifted from client-server to cloud, IoT and containerized applications. By eliminating the need for a central trust authority and the Public Key Infrastructure (PKI) model built 40 years ago for a client-server world, the new incubating project aims to provide a better framework for blockchain applications, cloud computing services, mobile and containerized developer applications.

The new Internet security framework works by establishing a new series of cryptographic service providers called Distributed Trust Authorities (DTAs) that independently issue shares of keys to application endpoints that have embedded Milagro cryptographic libraries and applications. In a DTA framework, the function of a pairing-based key generation server is split into three services, each of which issues thirds of private keys to distinct entities. Since key generation services are under separate organizational controls, current root key compromises and key escrow threats become an order of magnitude more difficult because an attacker would need to subvert all three independent parties to achieve the same result.

The project has the scope to expand for everyone, by creating a world without certificates, without passwords, and without single points of compromise. As an open source project, anyone can experiment with it, suggest changes and evaluate it for themselves. The following key components are available now, meaning that developers and security engineers can integrate with or build multi-factor authentication solutions into their existing Web properties or Web applications in minutes.

- The baseline **Milagro Crypto Library (MCL)** enables developers to build distributed trust systems and select from a choice of pairing-based protocols that deliver certificate-less key encapsulation, zero knowledge proof authentication, authenticated key agreement and digital signing functionality. Using MCL, application developers can embed multi-factor authentication, secure communications, and data protection methods that are robust enough to meet most requirements required by distributed ledger services, general on-line financial services, government and healthcare industries.
- **Milagro TLS**, a pairing-based TLS library, enables encrypted connections with perfect forward secrecy between mobile applications or IoT devices and backend infrastructures without the need for certificates or PKI. Milagro TLS is a standalone library that uses MCL as its cryptographic service provider, resulting in an implementation that is lean, yet high-performing enough to run in constrained environments found in many IoT devices.
- **Milagro MFA**, a multifactor authentication platform that uses zero knowledge proof protocols to eliminate the password and thus the threat of password database breach; Milagro MFA includes client SDKs in JavaScript, C, iOS, Android and Windows Phone, as well as the Authentication Server for Linux. Delivering 128-bit security but lean enough to even run in JavaScript, Milagro MFA allows developers and security engineers to integrate easy-to-use multi-factor authentication capabilities into their mobile and web properties and applications in hours or less.

Renewing trust in the Internet is a herculean task and something that we can't do alone. Trust, by nature, should be based on communities rather than individuals. That's why this project is a community-led, community-built program that relies on the participation and contributions of us-

ers. To find out more about the Apache Milagro (incubating) project, and download the code components, please visit the Milagro project page: <http://milagro.incubator.apache.org>



## About the Author:

We'd love to know what you think, on Twitter: [@apachemilagro](https://twitter.com/apachemilagro) or email: [https://www.miracl.com/about\\_us](https://www.miracl.com/about_us)

MIRACL is a leading internet cyber-security company that enables FORTUNE 2000 companies to remove their single largest security threat (the password database) as well as the roadblocks that hold back digital business transformation. MIRACL's Zero-Factor Authentication™ platform does not store passwords or PINs, thereby eliminating the credential theft attacks that organizations face today while enabling the opportunities of tomorrow.

MIRACL is also a co-founder of Apache Milagro (incubating) an open source initiative that enables decentralization technologies to improve security for people, apps and things. All MIRACL source code is made available through the Apache Milagro (incubating) project at [milagro.incubator.apache.org](http://milagro.incubator.apache.org). MIRACL is headquartered in London, with offices in San Francisco, and Tokyo.



# FREENAS MINI STORAGE APPLIANCE

IT SAVES YOUR LIFE.

## HOW IMPORTANT IS YOUR DATA?

Years of family photos. Your entire music and movie collection. Office documents you've put hours of work into. Backups for every computer you own. We ask again, *how important is your data?*

## NOW IMAGINE LOSING IT ALL

Losing one bit - that's all it takes. One single bit, and your file is gone.

The worst part? **You won't know until you absolutely need that file again.**

## THE SOLUTION

The FreeNAS Mini has emerged as the clear choice to save your digital life. **No other NAS in its class offers ECC (error correcting code) memory and ZFS bitrot protection to ensure data always reaches disk without corruption and *never degrades over time.***

No other NAS combines the inherent data integrity and security of the ZFS filesystem with fast on-disk encryption. No other NAS provides comparable power and flexibility. The FreeNAS Mini is, hands-down, the best home and small office storage appliance you can buy on the market. **When it comes to saving your important data, there simply is no other solution.**



*Example of one-bit corruption*

### The Mini boasts these state-of-the-art features:

- 8-core 2.4GHz Intel® Atom™ processor
- Up to 16TB of storage capacity
- 16GB of ECC memory (with the option to upgrade to 32GB)
- 2 x 1 Gigabit network controllers
- Remote management port (IPMI)
- Tool-less design; hot swappable drive trays
- FreeNAS installed and configured



<http://www.iXsystems.com/mini>



# FREENAS CERTIFIED STORAGE



With over six million downloads, FreeNAS is undisputedly *the* most popular storage operating system in the world.

Sure, you could build your own FreeNAS system: research every hardware option, order all the parts, wait for everything to ship and arrive, vent at customer service because it *hasn't*, and finally build it yourself while hoping everything fits - only to install the software and discover that the system you spent *days* agonizing over **isn't even compatible**. Or...

## MAKE IT EASY ON YOURSELF

As the sponsors and lead developers of the FreeNAS project, iXsystems has combined over 20 years of hardware experience with our FreeNAS expertise to bring you FreeNAS Certified Storage. **We make it easy to enjoy all the benefits of FreeNAS without the headache of building, setting up, configuring, and supporting it yourself.** As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS.

## Every FreeNAS server we ship is...

- » Custom built and optimized for your use case
- » Installed, configured, tested, and guaranteed to work out of the box
- » Supported by the Silicon Valley team that designed and built it
- » Backed by a 3 years parts and labor limited warranty

As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS. **Contact us today for a FREE Risk Elimination Consultation with one of our FreeNAS experts.** Remember, every purchase directly supports the FreeNAS project so we can continue adding features and improvements to the software for years to come. **And really - why would you buy a FreeNAS server from *anyone* else?**



### FreeNAS 1U

- Intel® Xeon® Processor E3-1200v2 Family
- Up to 16TB of storage capacity
- 16GB ECC memory (upgradable to 32GB)
- 2 x 10/100/1000 Gigabit Ethernet controllers
- Redundant power supply

### FreeNAS 2U

- 2x Intel® Xeon® Processors E5-2600v2 Family
- Up to 48TB of storage capacity
- 32GB ECC memory (upgradable to 128GB)
- 4 x 1GbE Network interface (Onboard) - (Upgradable to 2 x 10 Gigabit Interface)
- Redundant Power Supply

<http://www.ixsystems.com/storage/freenas-certified-storage/>



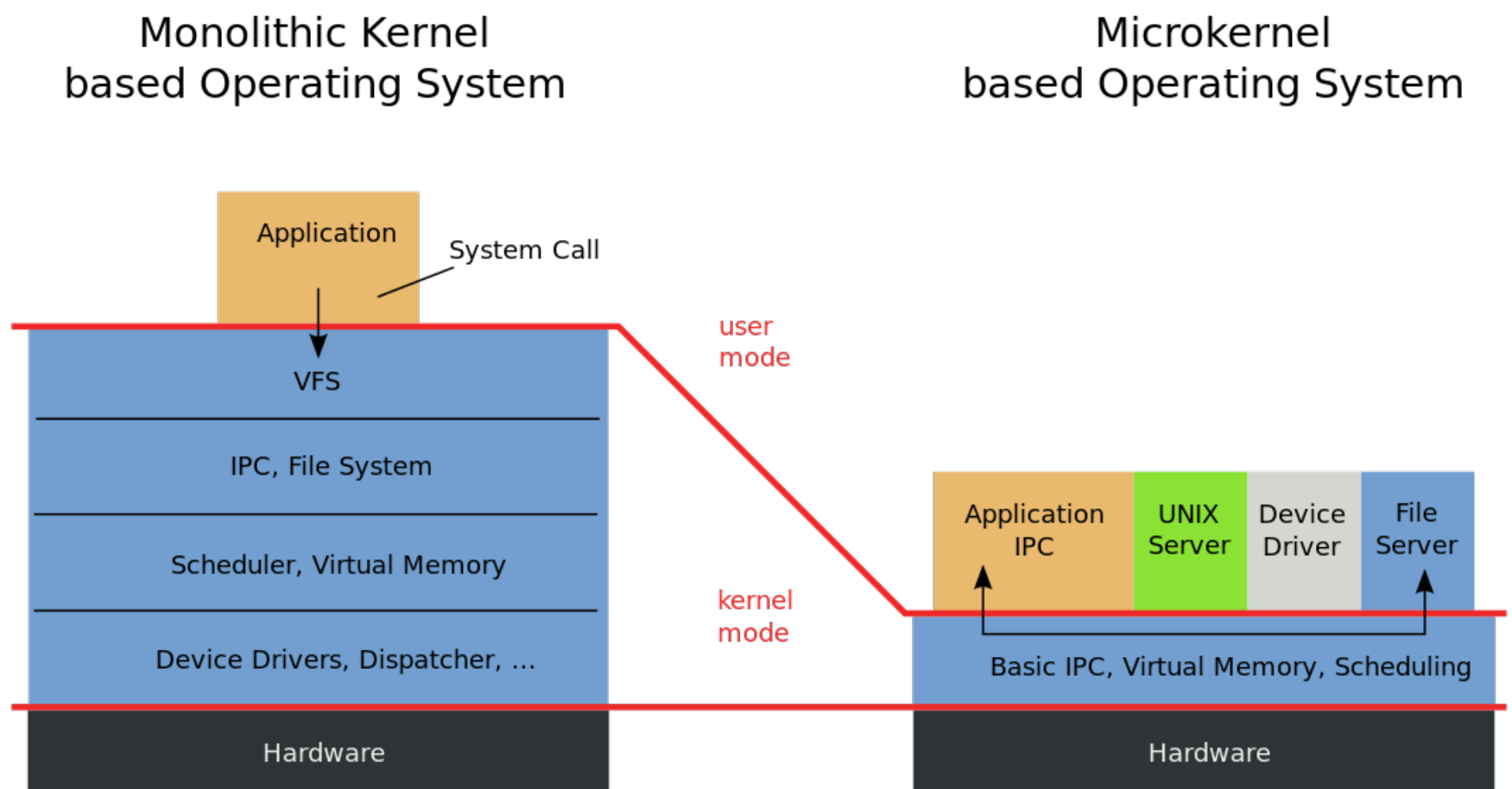


# bhyve: Introduction to Hypervisors

by Abdorrahman Homaei

A hypervisor or virtual machine monitor (VMM) is a computer software, firmware or hardware that creates and runs virtual machines. Actually, the power of VMM depends on the kernel model of the operating system. In general, there are three types of kernel model, microkernel, monolithic and hybrid. Here are pros and cons to each type..

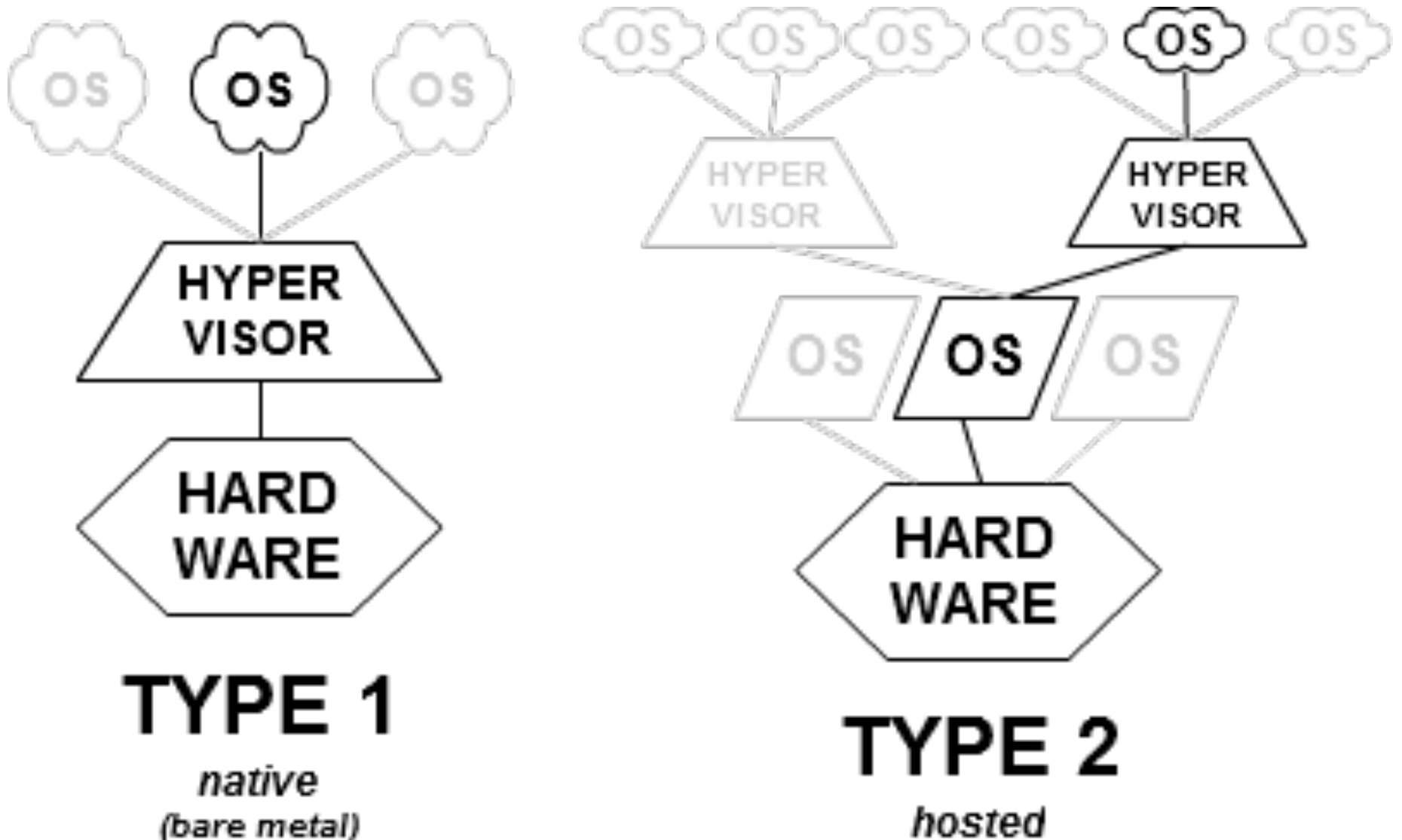
Kernel model concept is beyond the scope of this article and heavily needs OS knowledge but most hypervisors use two of them, microkernel and monolithic.



Microkernel kernels are slower but are more secure and stable and extendable. It means that adding a new feature to a monolithic kernel means recompiling the whole kernel, whereas with microkernels you can add new features or patches without recompiling.

Bear in mind that FreeBSD is not pure monolithic, it is modular monolithic and can load any driver dynamically.

There are also two types of VMMs:



**Type-1**, native or bare-metal hypervisors

These hypervisors run directly on the host's hardware to control the hardware and to manage guest operating systems. For this reason, they are sometimes called bare metal hypervisors. The first hypervisors, which IBM developed in the 1960s, were native hypervisors.

Today there are many type-1 hypervisors, like Citrix XenServer, Microsoft Hyper-V, and VMware ESX/ESXi.

Type-1 hypervisors can be monolithic or microkernel. For instance, Hyper-V is microkernel and ESXi is monolithic.

In fact, there is a controlling function that controls all aspects of the hypervisor. Hyper-V implements the controlling function in its Windows OS and in ESXi, the controlling function is imple-



mented within the ESXi kernel.

## **Type-2** or hosted hypervisors

A guest operating system runs as a process on the host. Type-2 hypervisors abstract guest operating systems from the host operating system. bhyve and kvm are in this domain.

It is difficult to say which design is better. However, there are a few advantages and disadvantages associated with each of them. One of the advantages of using the microkernelized type-1 design is that you can assign different roles to your hypervisor, like DNS or web-server, but on the other hand in this design, the system suffers from the lack of modern features, like a modern file-system.

In fact, performance and compatibility are not the only issue. In almost the same situation, simplicity is more valuable. If you want to easily combine hypervisor with something like zfs or carp, ignoring bhyve is so difficult.

## **What is Bhyve?**

bhyve (pronounced "bee hive", formerly written as BHyVe) is a type-2 hypervisor/virtual machine manager for FreeBSD that was introduced in FreeBSD 10.0 and supports most Intel and AMD processors that report the "POPCNT" (POPulation Count) processor feature in dmesg(8).

The bhyve BSD-licensed hypervisor became part of the base system with FreeBSD 10.0-RELEASE. This hypervisor supports a number of guests, including FreeBSD, OpenBSD, and many Linux distributions. Currently, bhyve only supports a serial console and does not emulate a graphical console. Virtualization offload features of newer CPUs are used to avoid the legacy methods of translating instructions and manually managing memory mappings.

The bhyve design requires a processor that supports Intel Extended Page Tables (EPT) or AMD Rapid Virtualization Indexing (RVI) or Nested Page Tables (NPT)

It runs FreeBSD 9+, OpenBSD, NetBSD, Linux and MS Windows desktop (versions Vista, 7, 8/8.1/8.2 and 10), as well as MS Windows Server (versions 2008/2008R2, 2012/2012R2 and 2016 Technical Preview 2 and 3) guests.

Lately, libvirt supports bhyve as well, but personally, I prefer to utilize bhyve from shell. There are also FreeBSD packages that were created to make life easier, like CBSD and VM-Bhyve.

Recently, the bhyve hypervisor supports Unified Extensible Firmware Interface Graphics Output Protocol or "UEFI-GOP". It means that you can easily run any modern OS without pain.

## Bhyve Configuration

The first step to creating a virtual machine in bhyve is configuring the host system. First, load the bhyve kernel module:

```
#kldload vmm
```

Then, create a tap interface for the network device in the virtual machine to attach to. In order for the network device to participate in the network, also create a bridge interface containing the tap interface and the physical interface as members. In this example, the physical interface is igb0:

```
# ifconfig tap0 create
# sysctl net.link.tap.up_on_open=1
net.link.tap.up_on_open: 0 -> 1
# ifconfig bridge0 create
# ifconfig bridge0 addm igb0 addm tap0
# ifconfig bridge0 up
```

## Creating a FreeBSD Guest

Create a file to use as the virtual disk for the guest machine. Specify the size and name of the virtual disk:

```
# truncate -s 16G guest.img
```

Download an installation image of FreeBSD to install:

```
# fetch
ftp://ftp.freebsd.org/pub/FreeBSD/releases/ISO-IMAGES/10.3/FreeBSD-10.3-RELEASE-amd64-bootonly.iso

FreeBSD-10.3-RELEASE-amd64-bootonly.iso      100% of 230 MB  570
kBps 06m17s
```

FreeBSD comes with an example script for running a virtual machine in bhyve. The script will start the virtual machine and run it in a loop, so it will automatically restart if it crashes. The script

The script will start the virtual machine and run it in a loop, so it will automatically restart if it crashes. The script takes a number of options to control the configuration of the machine: `-c` controls the number of virtual CPUs, `-m` limits the amount of memory available to the guest, `-t` defines which tap device to use, `-d` indicates which disk image to use, `-i` tells bhyve to boot from the CD image instead of the disk, and `-I` defines which CD image to use. The last parameter is the name of the virtual machine, used to track the running machines. This example starts the virtual machine in installation mode:

```
# sh /usr/share/examples/bhyve/vmrun.sh -c 4 -m 1024M -t tap0 -d
guest.img -i -I FreeBSD-10.3-RELEASE-amd64-bootonly.iso guestname
```

The virtual machine will boot and start the installer. After installing a system in the virtual machine, when the system asks about dropping in to a shell at the end of the installation, choose Yes. A small change needs to be made to make the system start with a serial console. Edit `/etc/ttys` and replace the existing `ttyu0` line with:

```
ttyu0    "/usr/libexec/getty 3wire"    xterm    on secure
```

Reboot the virtual machine. While rebooting the virtual machine causes bhyve to exit, the `vmrun.sh` script runs bhyve in a loop and will automatically restart it. When this happens, choose the reboot option from the boot loader menu in order to escape the loop. Now the guest can be started from the virtual disk:

```
# sh /usr/share/examples/bhyve/vmrun.sh -c 4 -m 1024M -t tap0 -d
guest.img guestname
```

## Creating a Linux Guest

Starting a virtual machine with bhyve is a two step process. First a kernel must be loaded, then the guest can be started. The Linux kernel is loaded with `sysutils/grub2-bhyve`. Create a `device.map` that grub will use to map the virtual devices to the files on the host system:

```
(hd0) ./linux.img
(cd0) ./somelinux.iso
```

Use `sysutils/grub2-bhyve` to load the Linux kernel from the ISO image:

```
# grub-bhyve -m device.map -r cd0 -M 1024M linuxguest
```

This will start grub. If the installation CD contains a `grub.cfg`, a menu will be displayed. If not, the `vmlinuz` and `initrd` files must be located and loaded manually:

```
grub> ls

(hd0) (cd0) (cd0,msdos1) (host)

grub> ls (cd0)/isolinux

boot.cat boot.msg grub.conf initrd.img isolinux.bin isolinux.cfg mem-
test

splash.jpg TRANS.TBL vesamenu.c32 vmlinuz

grub> linux (cd0)/isolinux/vmlinuz

grub> initrd (cd0)/isolinux/initrd.img

grub> boot
```

Now that the Linux kernel is loaded, the guest can be started:

```
# bhyve -A -H -P -s 0:0,hostbridge -s 1:0,lpc -s 2:0,virtio-net,tap1
-s 3:0,virtio-blk,./linux.img \
```

The system will boot and start the installer. After installing a system in the virtual machine, reboot the virtual machine. This will cause bhyve to exit. The instance of the virtual machine needs to be destroyed before it can be started again:

```
# bhyectl --destroy --vm=linuxguest
```

Now the guest can be started directly from the virtual disk. Load the kernel:

```
# grub-bhyve -m device.map -r hd0,msdos1 -M 1024M linuxguest

grub> ls
```



```
(hd0) (hd0,msdos2) (hd0,msdos1) (cd0) (cd0,msdos1) (host)

(lvm/VolGroup-lv_swap) (lvm/VolGroup-lv_root)

grub> ls (hd0,msdos1)/

lost+found/ grub/ efi/ System.map-2.6.32-431.el6.x86_64
config-2.6.32-431.el6.x
86_64 symvers-2.6.32-431.el6.x86_64.gz vmlinuz-2.6.32-431.el6.x86_64
initramfs-2.6.32-431.el6.x86_64.img

grub> linux (hd0,msdos1)/vmlinuz-2.6.32-431.el6.x86_64 root=/dev-
/mapper/VolGroup-lv_root

grub> initrd (hd0,msdos1)/initramfs-2.6.32-431.el6.x86_64.img

grub> boot
```

Boot the virtual machine:

```
# bhyve -A -H -P -s 0:0,hostbridge -s 1:0,lpc -s 2:0,virtio-net,tap1
\
-s 3:0,virtio-blk,./linux.img -l com1,stdio -c 4 -m 1024M linux-
guest
```

Linux will now boot in the virtual machine and eventually present you with the login prompt. Login and use the virtual machine. When you are finished, reboot the virtual machine to exit bhyve. Destroy the virtual machine instance:

```
# bhyvectl --destroy --vm=linuxguest
```

## Creating a Windows Guest

Now bhyve supports UEFI-GOP in FreeBSD 11.0-RELEASE and it is not required to “Remaster” Windows ISO anymore but you can also create a custom Windows ISO to install Windows in unattended-mode. A detailed description about “Remastering” Windows ISO is available at:

<http://pr1ntf.xyz/windowsunderbhyve.html>

## 1. Install FreeBSD 11.0

You can also install FreeBSD 11.0 or any of the latest builds.

## 2. Retrieve the firmware binary

We must to install “bhyve-firmware”. The best way to achieve this goal is to install with port mechanism. This process is very time-consuming and requires a lot of user-interaction, but with some tricks, we do it very easy:

```
# cd /usr/ports/sysutils/bhyve-firmware  
  
# make install clean -DBATCH
```

-DBATCH force port building process to not prompt you for confirmation and do it automatically.

## 3. Hypervisor, Network and Storage Preparation

```
# kldload vmm
```

this command will load bhyve kernel module or driver.

```
# ifconfig tap0 create up
```

this command creates a new interface and make it up.

```
# ifconfig bridge0 create up
```

this command also creates a bridge and make it up and ready.

```
# ifconfig bridge0 addm em0
```

this command adds em0 (network interface) to bridge0.

```
# ifconfig bridge0 addm tap0
```

this command adds tap0 to bridge0.

```
# truncate -s 50G disk.img
```

this command creates a file with 50GB size.

#### 4. Boot a Virtual Machine:

```
# bhyve -c 2 -m 4G -w -H \
-s 0,hostbridge \
-s 3,ahci-cd,/path/to/windows-2012R2.iso \
-s 4,ahci-hd,disk.img \
-s 5,virtio-net,tap0 \
-s 29,fbuf,tcp=0.0.0.0:5900,w=800,h=600,wait \
-s 30,xhci,tablet \
-s 31,lpc -l com1,stdio \
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_UEFI.fd \
vm0
```

his command makes a virtual machine(vm0) with cores CPU and with a display resolution of 800 by 600 that can be accessed via VNC at: 0.0.0.0:5900

The fbuf wait parameter instructs bhyve to only boot upon the initiation of a VNC connection, simplifying the installation of operating systems that require immediate keyboard input. This can be removed for post-installation use.

The xhci,tablet parameter provides precise cursor synchronization when using VNC, but is not supported by FreeBSD.

Desktop versions of Microsoft Windows require the presence of a CD/DVD device, which can be an empty file created with touch(1).

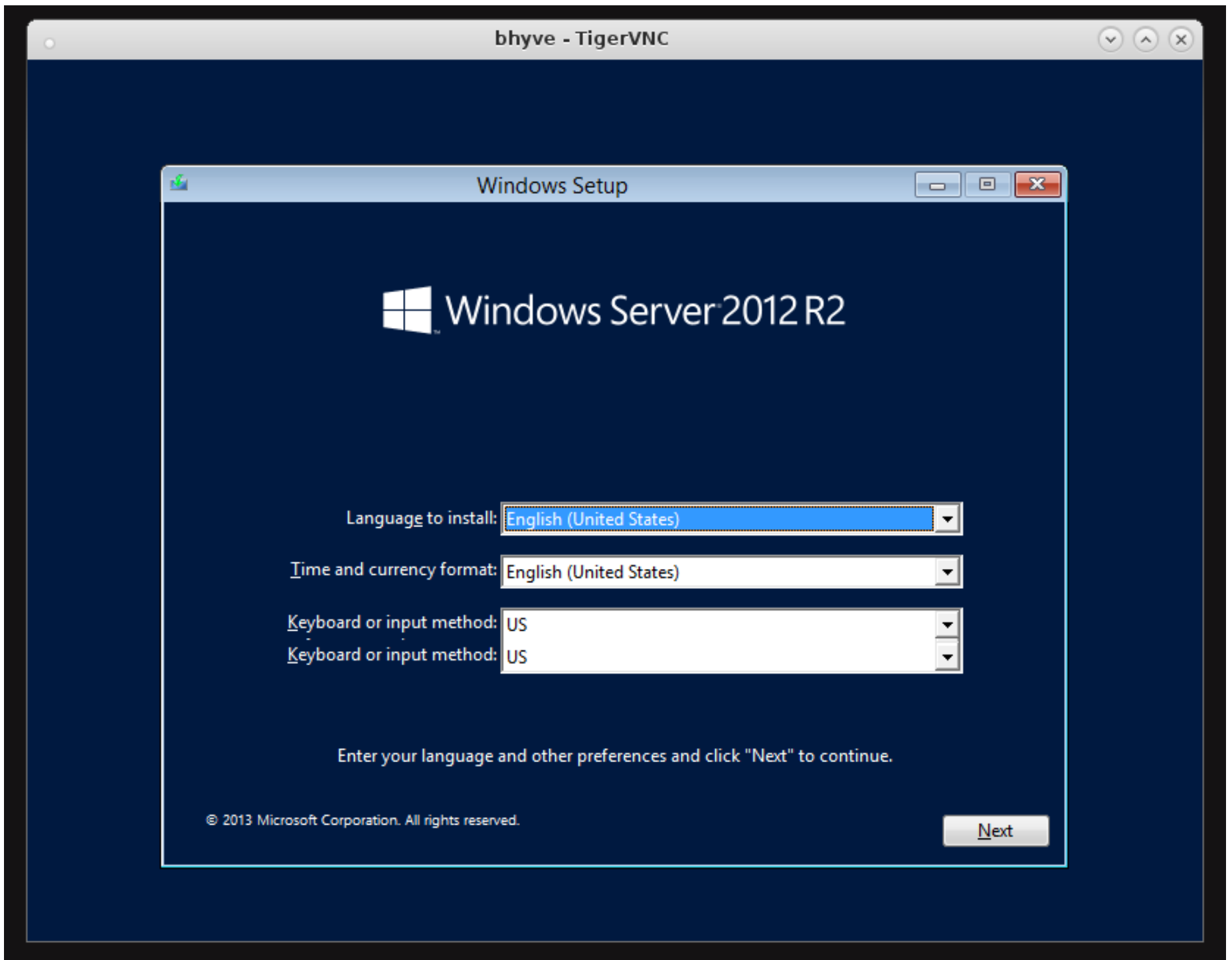
-H Yield the virtual CPU thread when a HLT instruction is detected. If this option is not specified, virtual CPUs will use 100% of a host CPU.

-w Ignore accesses to unimplemented Model Specific Registers (MSRs). This is intended for debug purposes.

## 5. Connect to VM with VNC client

```
# vncviewer 192.168.1.1:5900
```

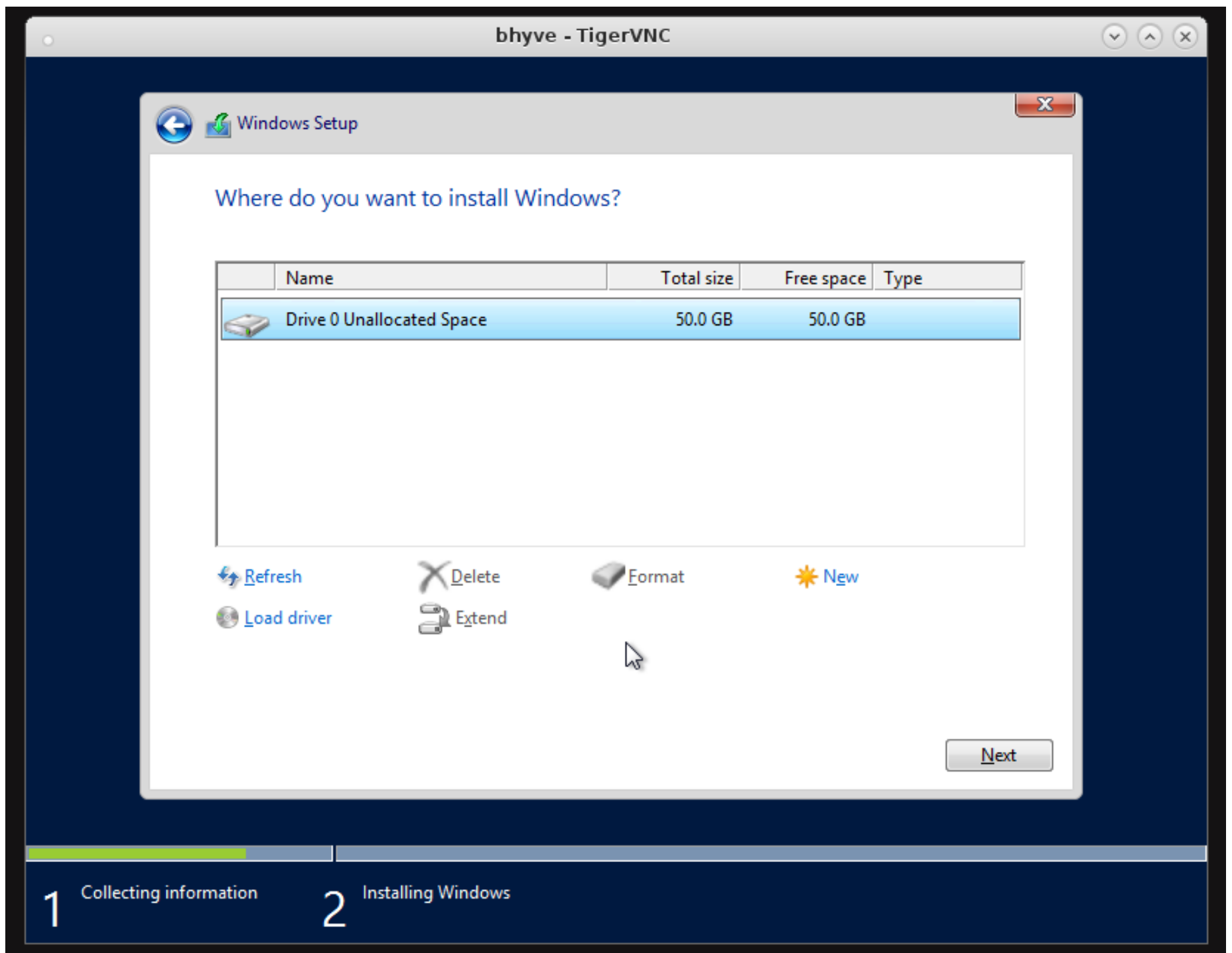
In VNC Client screen, you can see what is happening, also mice are supported. I prefer to use “tightvnc”. My hypervisor IP is “192.168.1.1” and I have a DHCP on my network so Windows gets the IP address automatically.



## 6. Setup Process

Setup process needs to restart vm. After each restart, you must run it again until the setup completion.





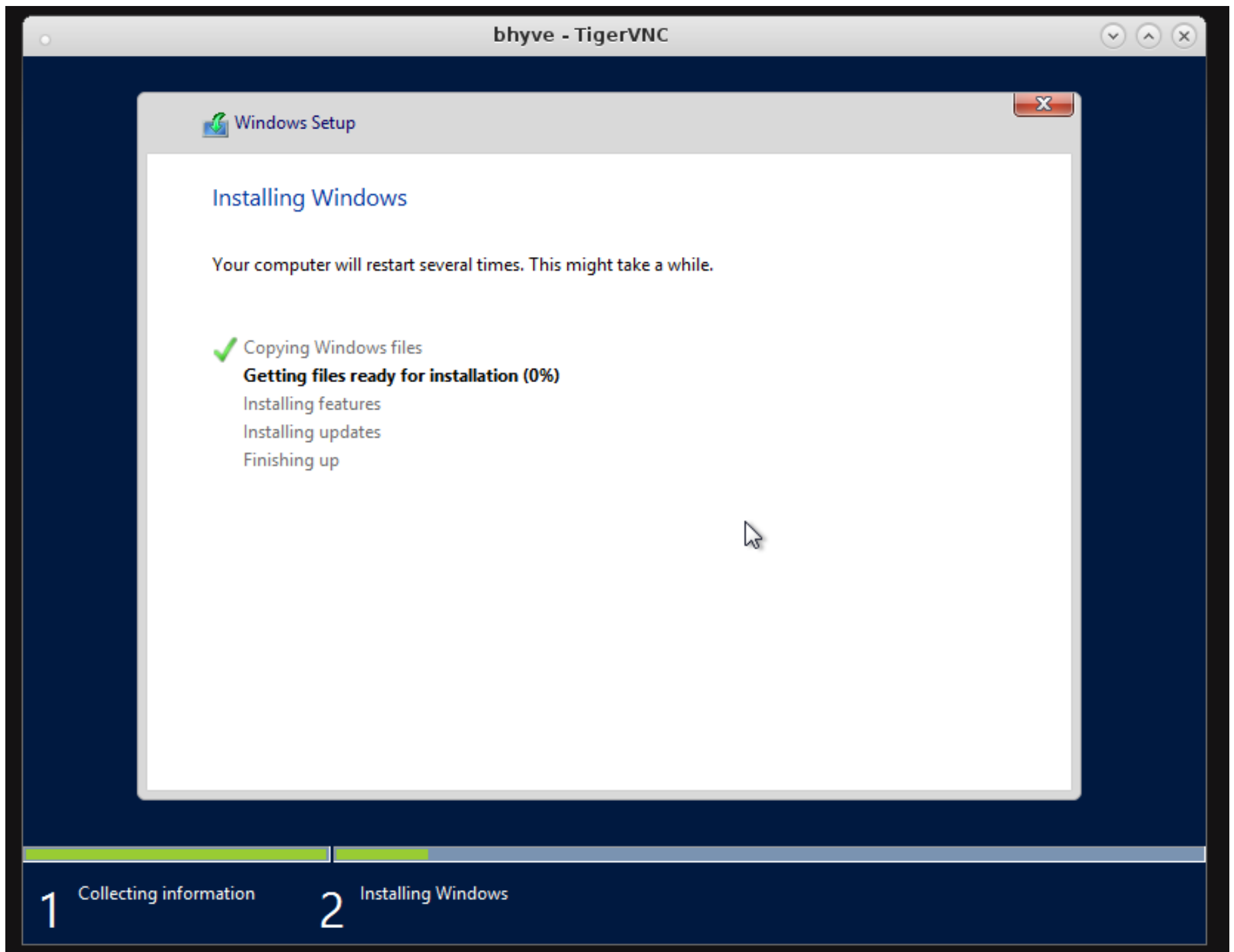
7. Virtio is a virtualization standard for network and disk device drivers where just the guest's device driver "knows" it is running in a virtual environment, and cooperates with the hypervisor. This enables guests to get high performance network and disk operations, and gives most of the performance benefits of paravirtualization.

Virtio can be downloaded from below link:

<https://fedorapeople.org/groups/virt/virtio-win/direct-downloads/archive-virtio/virtio-win-0.1.118-2/virtio-win-0.1.118.iso>

## 8. Setup NIC Driver

After first login, you must shutdown the vm and issue this command:



```
# bhyvectl --destroy --vm=vm0

# bhyve -c 2 -m 4G -w -H \
-s 0,hostbridge \
-s 3,ahci-cd,/path/to/virtio-win-0.1.118.iso \
-s 4,ahci-hd,disk.img \
```

```
-s 5,virtio-net,tap0 \  
-s 29,fbuf,tcp=0.0.0.0:5900,w=800,h=600,wait \  
-s 30,xhci,tablet \  
-s 31,lpc -l com1,stdio \  
-l bootrom,/usr/local/share/uefi-firmware/BHYVE_UEFI.fd \  
vm0
```

After logging in to your desktop, you can easily find the proper driver and install it. Then you can set IP and connect to your vm with remote desktop client. rdp is much faster than vnc.

## Conclusion

As you can see, running virtual machine under FreeBSD's bhyve is very easy. I also tested bhyve with some gstrip RAID from five SSD hard disks with 120 GB size. Now I can boot Windows or Linux in seven seconds.

FreeBSD's focus on performance, networking, and storage combines with ease of system administration and comprehensive documentation to realize the full potential of any computer.

## Bhyvecon

Bhyvecon is a conference dedicated to BSD hypervisors. The third annual bhyvecon Tokyo will take place on March 11th, 2016 from 16:15 to 21:30 at the Tokyo University of Science after the BSD Vendor Summit.

---

## Useful Links

For a complete list of Intel processors that support EPT:

<http://ark.intel.com/search/advanced?s=t&ExtendedPageTables=true>

Management system for bhyve virtual machines:

<https://github.com/churchers/vm-bhyve>

Bhyve Website:

<http://www.bhyve.org>

Tutorial:

<http://pr1ntf.xyz/windowsunderbhyve.html>

---



### About the Author:

Abdorrahman Homaei has been working as a software developer since 2000.

He used FreeBSD for more than 10 years. He became involved with the meetBSD dot ir and performed serious training on FreeBSD. He worked for a mobile network operator (mci) company that provides sim card for 24 years in Iran. He also used FreeBSD for penetration testing.



## How to Install and Configure VNC on Ubuntu 16.04

*by Moustafa N. El-Zeny*

**Odoo is a web-based Open Source enterprise resource planning and customer relationship software that helps to organize and grow your business. Odoo was formerly known as OpenERP and, before that, TinyERP. There are many apps available to extend Odoo, for example: billing, accounting, manufacturing, purchasing, warehouse management, and project Management.**

---

### Prerequisites

- Ubuntu 16.04 - 64bit
- 2GB memory.

### What we will do in this tutorial:

- Add the Odoo repository.
- Configure a Linux user for Odoo.
- Install and Configure PostgreSQL.
- Install dependencies needed by Odoo.
- Install Odoo.
- Configure Odoo.

---

Odoo was created by Fabien Pinckaers/Odoo S.A and written in python. Currently, it's available and compatible with many operating systems, including Linux, Windows and Mac OS X. For server installation, I will use Ubuntu 16.04. Odoo released version 9 of their ERP software on October 1, 2015.

## Step 1 - Add the Odoo repository.

First, you will have to add the Odoo apt repository to your repository database file `/etc/apt/sources.list`. In order to do this, add the Odoo key using this command:

```
wget -O - https://nightly.odoo.com/odoo.key | apt-key add -
```

Secondly, add the Odoo repository using the echo command:

```
echo "deb http://nightly.odoo.com/8.0/nightly/deb/ ." >>
```

**Note:** >> = Add the Odoo repository to the last line in `sources.list` file and update the Ubuntu package lists:

```
apt-get update
```

## Step 2 - Configure a Linux user for Odoo.

First, create a new user called `odoo` with home directory `/opt/odoo` and the group `odoo`. You can do this using this command:

```
sudo adduser --system --home=/opt/odoo --group odoo
```

Thereafter, create a new directory for Odoo in the `/var/lib/` directory.

```
mkdir -p /var/lib/odoo
```

## Step 3 - Install and Configure PostgreSQL.

You can install PostgreSQL with this `apt-get` command:

```
sudo apt-get install postgresql
```

Afterwards, log into the PostgreSQL shell:

```
su - postgres
```

Now, create a role for Odoo. This will allow Odoo to access or connect to the PostgreSQL server and to create, delete or modify the database. You will have to enter your password for security reasons and also to have it private.

```
createuser --createdb --username postgres --no-createrole --no-
superuser -- pwprompt odoo
```

Finally, you can log out from the PostgreSQL shell by typing `exit`.

## Step 4 - Install dependencies needed by Odoo.

Odoo requires many python modules. To achieve this, just install all the packages below to fulfill its prerequisites:

```
sudo apt-get install python-cups python-dateutil python-decorator
python-gdata python-geoip python-gevent python-imaging python-jinja2
python-lxml python-mako python-mock python-openid python-passlib
python-pybabel python-pychart python-pydot python-pyparsing
python-simplejson python-tz python-unicodcsv python-unittest2
python-werkzeug python-xlwt python-yaml wkhtmltopdf
```

## Step 5 - Install Odoo:

Now, you can install Odoo with apt:

```
sudo apt-get install odoo
```

When the installation is complete, Odoo will be running on port 8069. You can check it using the command:

```
netstat -plntu
```

```
root@odoo:~# netstat -plntu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN      2332/sshd
tcp        0      0 127.0.0.1:5432         0.0.0.0:*               LISTEN      5483/postgres
tcp        0      0 0.0.0.0:8069           0.0.0.0:*               LISTEN      11676/python
tcp6       0      0 :::22                  :::*                     LISTEN      2332/sshd
udp        0      0 0.0.0.0:68             0.0.0.0:*               *          2639/dhclient
udp        0      0 0.0.0.0:68             0.0.0.0:*               *          2036/dhclient
root@odoo:~#
```

Once Odoo is installed, it runs on port 8069. We will run Odoo on local IP, because we will use aNginx web server as a reverse proxy for Odoo.

Edit the Odoo configuration file with vim command:

```
vim /etc/odoo/openerp-server.conf
```

At the end of the file, paste the following configuration:

```
xmlrpc_interface = 127.0.0.1  
  
xmlrpc_port = 8069
```

Lastly, save and exit.

## Step 6 – Access Odoo from Nginx Frontend.

You can configure the system such that users can access an Odoo web panel via Nginx reverse proxy. This will enable users to navigate the Odoo web interface faster, due to some Nginx frontend caching, on standard HTTP ports without the need to manually enter the http port 8069 on their browsers.

In order to configure this setting, you need to install and configure Nginx on your system by following these steps:

1. Install Nginx web server using the following command:

```
# apt-get install nginx
```

2. Next, open Nginx main configuration file with a text editor and insert the following block after the line which specifies Nginx document root location.

```
# nano /etc/nginx/sites-enabled/default
```

Add the following configuration excerpt to `nginx.conf` file:

```
location / {  
  
    proxy_pass http://127.0.0.1:8069;  
  
    proxy_redirect off;
```



```
proxy_set_header Host $host;

proxy_set_header X-Real-IP $remote_addr;

proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
```

Also, comment on Nginx **location** statement by placing a # in front of the following lines. Use the screenshot below as a guide.

```
#location / {

    • First attempt to serve request as file, then

    • as directory, then fall back to displaying a 404

    • try_files $uri $uri/ =404;

#}
```

```
GNU nano 2.3.1 File: /etc/nginx/nginx.conf

sendfile            on;
tcp_nopush          on;
tcp_nodelay         on;
keepalive_timeout   65;
types_hash_max_size 2048;

include             /etc/nginx/mime.types;
default_type        application/octet-stream;

# Load modular configuration files from the /etc/nginx/conf.d directory.
# See http://nginx.org/en/docs/nginx_core_module.html#include
# for more information.
include /etc/nginx/conf.d/*.conf;

server {
    listen      80 default_server;
    listen      [::]:80 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    location / {
        proxy_pass http://127.0.0.1:8069;
        proxy_redirect off;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    #location / {
    #}
}
```

[ line 54/67 (80%), col 1/1 (100%), char 1607/1821 (88%) ]

^G Get Help   ^C WriteOut   ^R Read File   ^Y Prev Page   ^K Cut Text   ^C Cur Pos  
^X Exit   ^J Justify   ^W Where Is   ^V Next Page   ^U UnCut Text   ^T To Spell

3. This step is a security optional feature and implies the change of the network socket that Odoo application is listening, changing the binding address from all interfaces (or address) to localhost only.

This change must only be done in conjunction with Nginx reverse proxy due to the fact that binding the application on a localhost implies that Odoo will not be accessible by users inside the LAN or other networks.

In order to effect this change, open `/etc/odoo/openerp-server.conf` file and edit `xmlrpc_interface` line to bind on localhost only as shown in the screenshot below.

```
xmlrpc_interface = 127.0.0.1
```

```
GNU nano 2.3.1 File: /etc/nginx/nginx.conf

sendfile            on;
tcp_nopush          on;
tcp_nodelay         on;
keepalive_timeout   65;
types_hash_max_size 2048;

include             /etc/nginx/mime.types;
default_type        application/octet-stream;

# Load modular configuration files from the /etc/nginx/conf.d directory.
# See http://nginx.org/en/docs/nginx_core_module.html#include
# for more information.
include /etc/nginx/conf.d/*.conf;

server {
    listen            80 default_server;
    listen            [::]:80 default_server;
    server_name       _;
    root              /usr/share/nginx/html;

    location / {
        proxy_pass    http://127.0.0.1:8069;
        proxy_redirect off;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    #location / {
    #}

[ line 54/67 (80%), col 1/1 (100%), char 1607/1821 (88%) ]
^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is     ^V Next Page     ^U UnCut Text    ^T To Spell
```

To reflect the changes, restart Odoo service by running the below command:

```
#systemctl restart odoo.service OR
```

```
# service odoo restart
```

4. In case your machine has a network defense line provided by the firewall, issue the following commands to open firewall ports to outside world for Nginx proxy:

```
# iptables -A INPUT -p tcp -m tcp --sport 80 -j ACCEPT
```

## Step 7 - Configure Odoo.

Now that both Odoo and Nginx are installed, open a web browser and type in the Odoo URL.

You will be prompted to create a new database, configure your email address and password for the admin account.

Proceed to enter the details for your installation and click on 'Create database'.



The screenshot shows the Odoo installation configuration page. At the top is the Odoo logo. Below it, a message says "Odoo is up and running! Create a new database by filling out the form, you'll be able to install your first app in a minute." The form contains the following fields:

- Database Name:** A text input field with the value "odoo\_db".
- Email:** A text input field with the value "admin@mysite.co".
- Password:** A password input field with masked characters and a toggle icon.
- Language:** A dropdown menu showing "English (US)".
- Country:** A dropdown menu showing "Indonesia".
- ☐ **Load demonstration data** (Check this box to evaluate Odoo)
- Create database:** A blue button at the bottom.

Wait patiently, about a minute, or the Odoo's installation to finish.

After Odoo has created a database, you can login to the Odoo admin dashboard. However, for security reason, you need to setup a master password for the Odoo database manager. Click on **'Manage Databases'**



Email

Password

Log in

[Manage Databases](#) | Powered by Odoo

Click on **'Set Master Password'**.



Warning, your Odoo database manager is not protected. Please [set a master password](#) to secure it.

odoo\_db

Backup

Duplicate

Delete

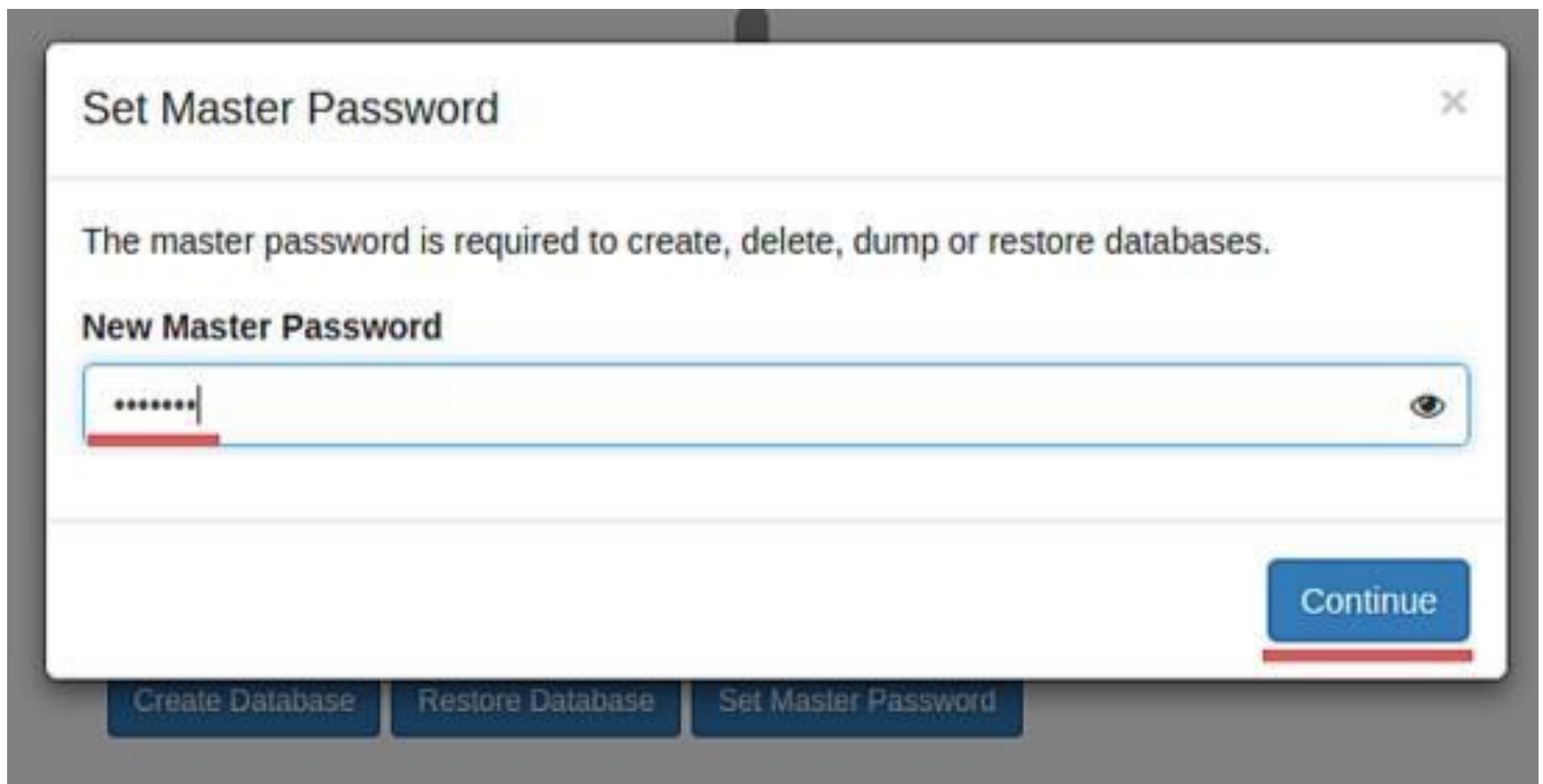
Create Database

Restore Database

[Set Master Password](#)



Type your password and click '**Continue**'.



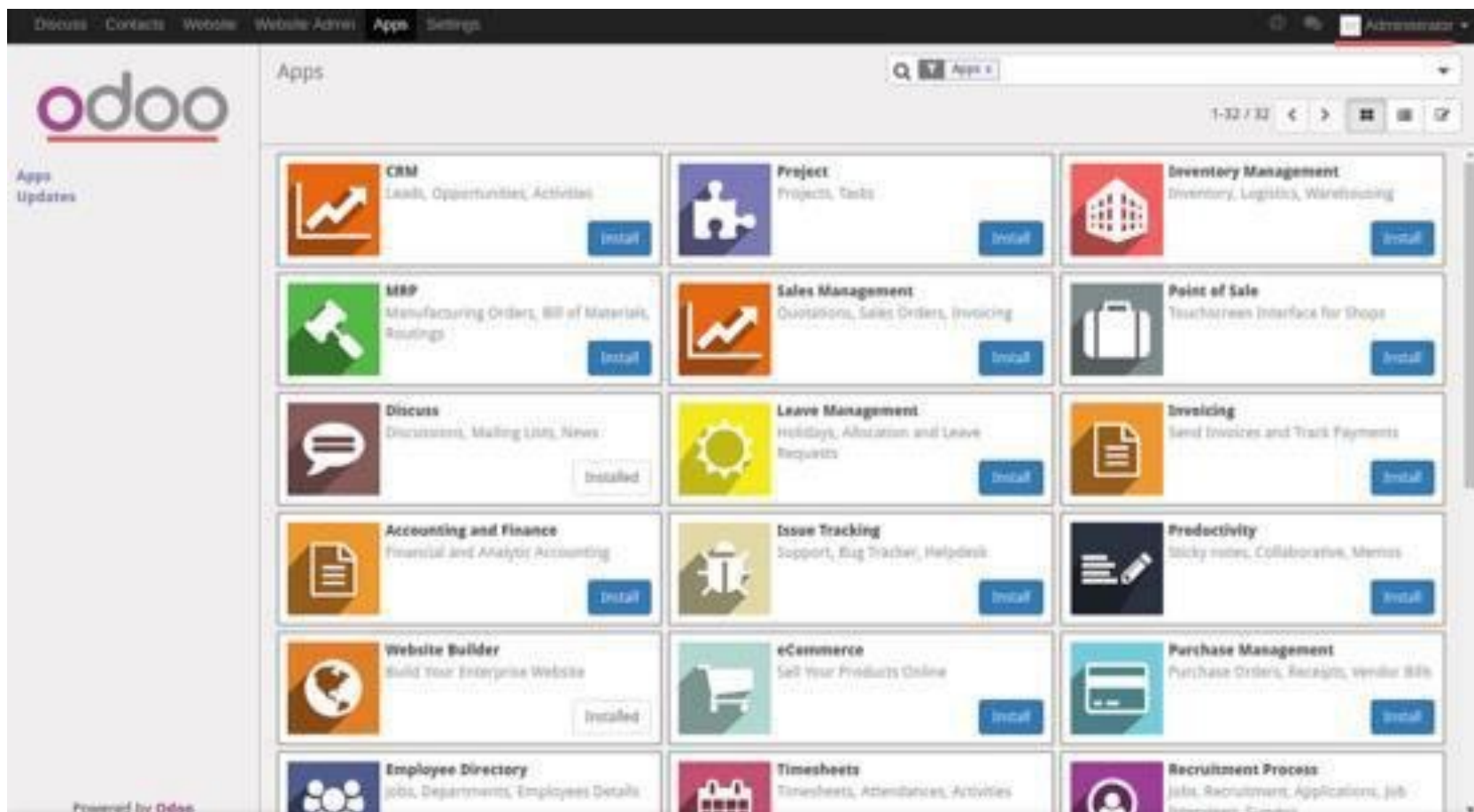
The screenshot shows a 'Set Master Password' dialog box. At the top, it says 'Set Master Password' with a close button (X) on the right. Below this, a message states: 'The master password is required to create, delete, dump or restore databases.' Underneath, there is a section titled 'New Master Password' with a text input field containing several dots (masked password) and a toggle icon (an eye) on the right. A blue 'Continue' button is located at the bottom right of the dialog. Below the dialog, three buttons are visible: 'Create Database', 'Restore Database', and 'Set Master Password'.

Type your email and password and click '**Login**'.

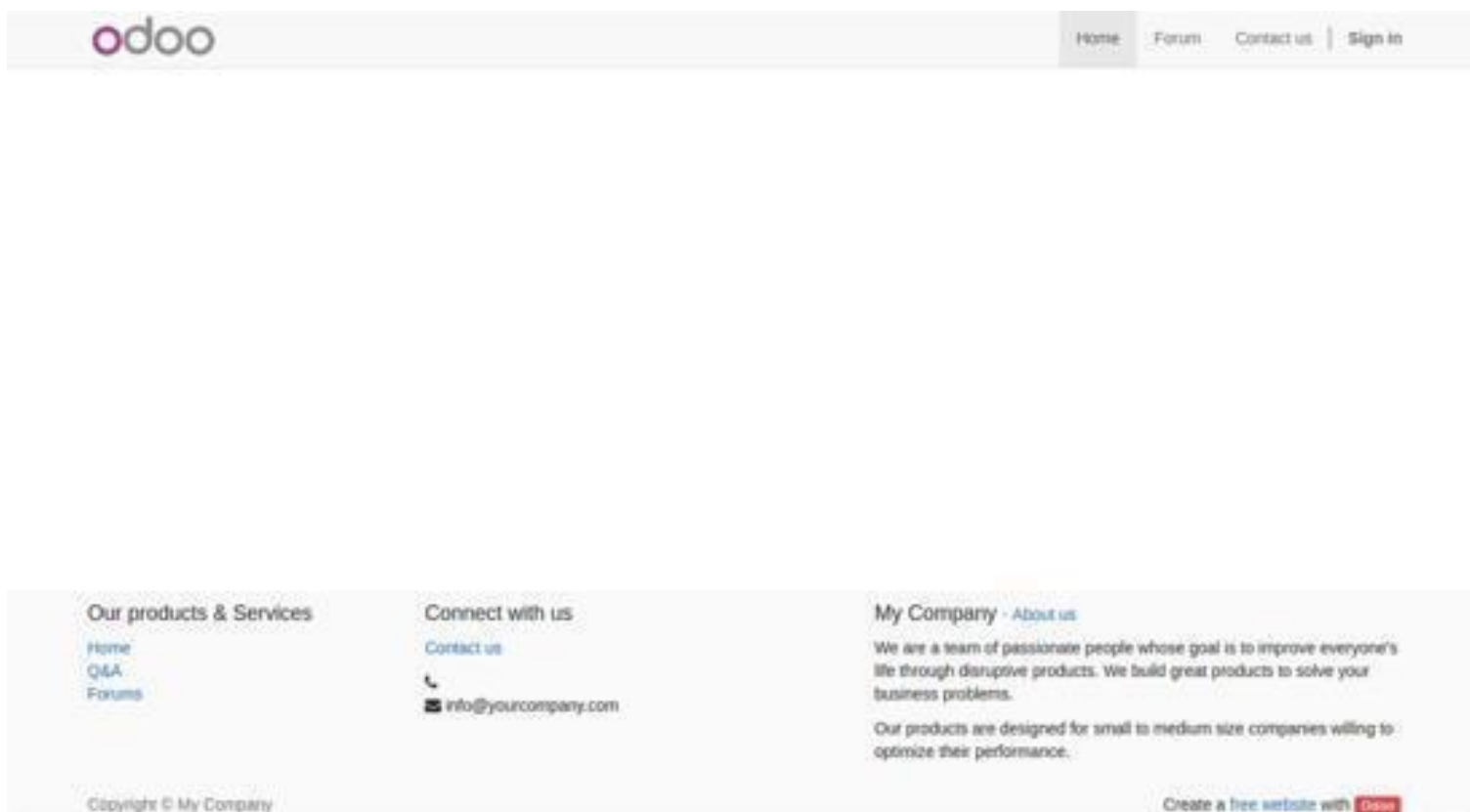


The screenshot shows the Odoo login page. At the top center is the 'odoo' logo. Below it, there are two input fields: 'Email' and 'Password'. The 'Email' field contains the text 'admin@mysite.co'. The 'Password' field contains several dots (masked password). Below the password field is a blue 'Log in' button. At the bottom of the page, there is a footer that reads 'Manage Databases | Powered by Odoo'.

Here is the Odoo admin dashboard.



Odoo home page after installing a new app called 'Forum'.



Odoo version 9 with Nginx has been successfully installed on Ubuntu 16.04.

## How to Install and Configure VNC on Ubuntu 16.04.04

VNC, or "Virtual Network Computing," is a connection system that allows you to use your keyboard and mouse to interact with a graphical desktop environment on a remote server. It makes managing files, software and settings on a remote server easier for users who are not yet comfortable with the command line.

In this guide, we will be setting up VNC on an Ubuntu 16.04 server and connect securely through an SSH tunnel. The VNC server we will be using is TightVNC, a fast and lightweight remote control package. This choice will ensure that our VNC connection is smooth and stable even on slower internet connections.

---

### To complete this tutorial, you'll need:

- An Ubuntu 16.04 Droplet.
- A local computer with a VNC client installed that supports VNC connections over SSH tunnels. If you are using Windows, you could use TightVNC, RealVNC or UltraVNC. Mac OS X users can use the built-in Screen Sharing program, or can use a cross-platform app like RealVNC. Linux users have many options: vinagre, krdc, RealVNC, TightVNC, and more.

---

### Step 1 — Installing the Desktop Environment and VNC Server.

By default, an Ubuntu 16.04 Droplet does not come with a graphical desktop environment or a VNC server installed. Thus, we'll begin by installing those. Specifically, we will install packages for the latest Xfce desktop environment and the TightVNC package available in the official Ubuntu repository.

On your server, install the Xfce and TightVNC packages by typing:

```
$sudo apt install xfce4 xfce4-goodies tightvncserver
```

To complete the VNC server's initial configuration after installation, use the `vncserver` command to set up a secure password.

```
$vncserver
```

You'll be prompted to enter and verify a password, and also, a view-only password. Users who log in with the view-only password will not be able to control the VNC instance with their mouse or keyboard. This is a helpful option if you want to demonstrate something to other people using your VNC server, but isn't necessary.

Running `vncserver` completes the installation of VNC by creating default configuration files and connection information for our server to use. With these packages installed, you are now ready to configure your VNC server.

## Step 2 — Configuring the VNC Server.

First, we need to tell our VNC server what commands to run when it starts up. These commands are located in a configuration file called `xstartup` in the `.vnc` folder under your home directory. The startup script was created when you ran the `vncserver` in the previous step, though, we need to modify some of the commands for the Xfce desktop.

When VNC is first set up, it launches a default server instance on port 5901. This port is called a display port and is referred to by VNC as 1. VNC can launch multiple instances on other display ports, like: 2, 3, etc. When working with VNC servers, remember that: X is a display port that refers to  $5900+X$ .

Because we are going to be changing how the VNC server is configured, we'll need first to stop the VNC server instance that is running on port 5901.

```
$vncserver -kill :1
```

The output should look like this, with a different PID:

Output

```
Killing Xtightvnc process ID 17648
```

Before we begin configuring the new `xstartup` file, let's back up the original.

```
$ mv ~/.vnc/xstartup ~/.vnc/xstartup.bak
```



Now create a new xstartup file with nano or your favorite text editor.

```
$ nano ~/.vnc/xstartup
```

Paste these commands into the file so that they are automatically executed whenever you start or restart the VNC server. Finally, save and close the file.

```
~/.vnc/xstartup

#!/bin/bash

xrdb $HOME/.Xresources

startxfce4 &
```

The first command in the file, `xrdb $HOME/.Xresources`, tells VNC's GUI framework to read the server user's `.Xresources` file. `.Xresources` is where a user can make changes to certain settings of the graphical desktop, like terminal colors, cursor themes and font rendering. The second command simply tells the server to launch Xfce, which is where you will find all of the graphical software that you need to manage your server comfortably.

To ensure that the VNC server will be able to use this new startup file properly, we'll need to grant executable privileges to it.

```
$ sudo chmod +x ~/.vnc/xstartup
```

Now, restart the VNC server.

```
$ vncserver
```

The server should be started with an output similar to this:

```
Output

New 'X' desktop is your_server_name.com:1

Starting applications specified in /home/sammy/.vnc/xstartup

Log file is /home/sammy/.vnc/liniverse.com:1.log
```

## Step 3 — Testing the VNC Desktop.

In this step, we'll test the connectivity of your VNC server.

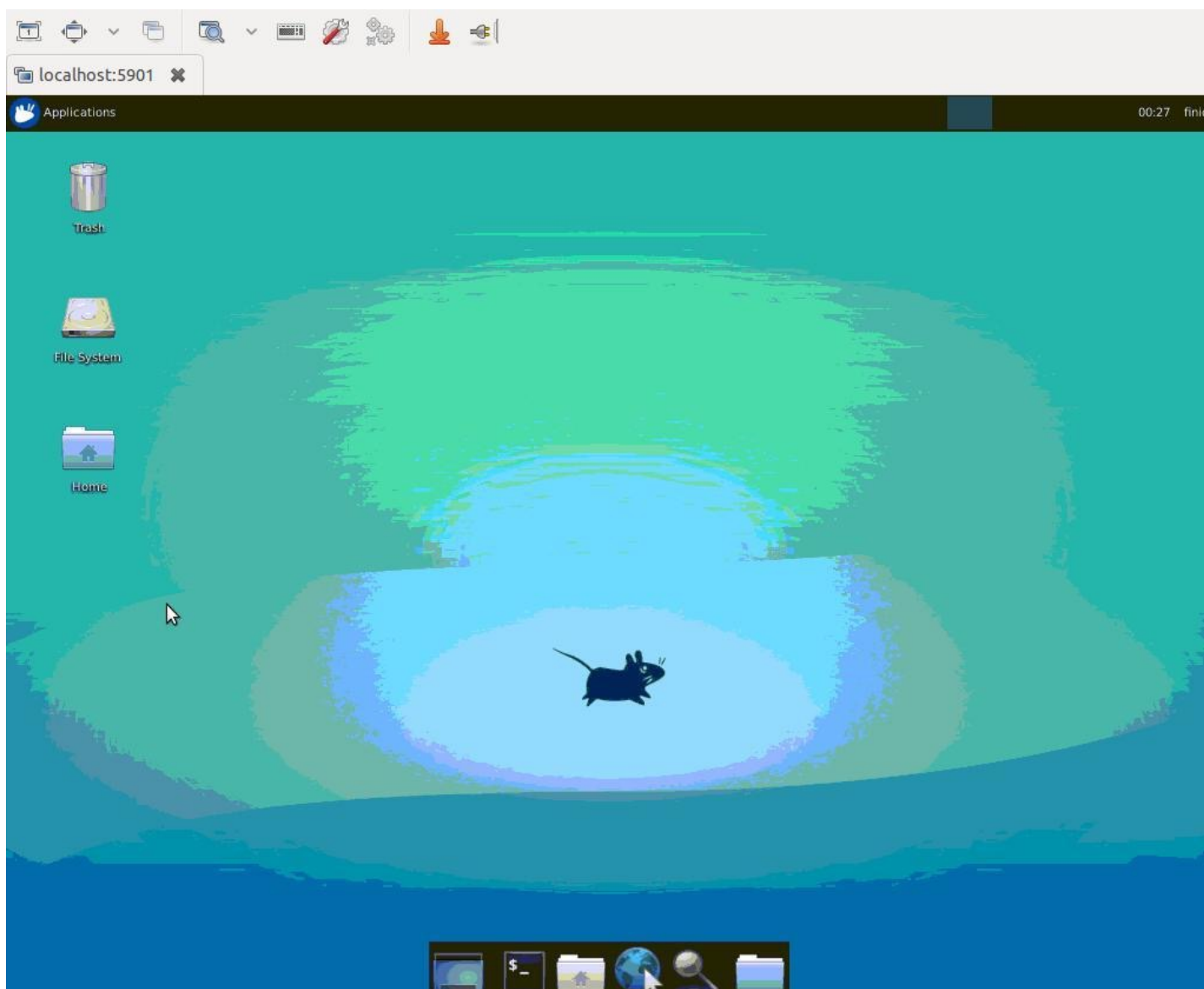
First, we'll need to create an SSH connection on your local computer that securely forwards to the localhost connection for VNC. You can do this via the terminal on Linux or OS X with the following command. Remember to replace `user` and `server_ip_address` with the sudo non-root username and IP address of your server.

```
$ ssh -L 5901:127.0.0.1:5901 -N -f -l username server_ip_address
```

If you are using a graphical SSH client, like PuTTY, use `server_ip_address` as the connection IP and `setlocalhost:5901` as a new forwarded port in the program's SSH tunnel settings.

Next, you may now use a VNC client to attempt a connection to the VNC server at `localhost:5901`. You'll be prompted to authenticate. The correct password to use is the one you provided in Step 1.

Once you are connected, you should see the default Xfce desktop. It should look something like this:



## Step 4 — Creating a VNC Service File

Next, we'll set up the VNC server as a systemd service. This will make it possible to start, stop and restart it as needed, like any other systemd service.

First, create a new unit file called `/etc/systemd/system/vncserver@.service` using your favorite text editor:

```
$ sudo nano /etc/systemd/system/vncserver@.service
```

Copy and paste the following into it. Be sure to change the value of `User` and the username in the value of **PIDFILE** to match your username.

```
/etc/systemd/system/vncserver@.service

[Unit]

Description=Start TightVNC server at startup

After=syslog.target network.target


[Service]

Type=forking

User=sammy

PAMName=login

PIDFile=/home/sammy/.vnc/%H:%i.pid ExecStartPre=-/usr/bin/vncserver
-kill :%i > /dev/null 2>&1 ExecStart=/usr/bin/vncserver -depth 24 -ge-
ometry 1280x800 :%i ExecStop=/usr/bin/vncserver -kill :%i


[Install]

WantedBy=multi-user.target
```

Save and close the file.

Next, make the system aware of the new unit file by typing this command.

```
$ sudo systemctl daemon-reload
```

Enable the unit file:

```
$ sudo systemctl enable vncserver@1.service
```

Stop the current instance of the VNC server if it's still running:

```
$ vncserver -kill :1
```

Then, start it as you would start any other systemd service:

```
$ sudo systemctl start vncserver@1
```

You can verify that it started using this command:

```
$ sudo systemctl status vncserver@1
```

If it started correctly, the output should look like this:

## Output

```
vncserver@1.service - TightVNC server on Ubuntu 16.04
```

```
Loaded: loaded (/etc/systemd/system/vncserver@.service; enabled; vendor preset: enabled)
```

```
Active: active (running) since Mon 2016-04-25 03:21:34 EDT; 6s ago
```

```
Process: 2924 ExecStop=/usr/bin/vncserver -kill
```

```
systemd[1]: Starting TightVNC server on Ubuntu 16.04...
```

```
systemd[2938]: pam_unix(login:session): session opened for user finid by (uid=0)
```

```
systemd[2949]: pam_unix(login:session): session opened for user finid by
```



by (uid=0)

```
systemd[1]: Started TightVNC server on Ubuntu 16.04.
```

## Conclusion

Odoo is an open source application that helps you to manage your business. Odoo / OpenERP is easy to install and configure and supports multiple operating systems. In Odoo, there are many applications available such as billing, accounting, manufacturing, purchasing, warehouse management and project management. These apps can help you to manage and grow your business.

You should now have a secured VNC server up and running on your Ubuntu 16.04 server. Now, you'll be able to manage your files, software and settings with an easy-to-use and a familiar graphical interface.



### About the Author:

This is Moustafa Nabil El-Zeny. I'm a Red Hat Geek, oldest Guru and an activist (since 2005) who is passionate about open-source. I graduated from Mansourah University, Faculty of Engineering, Computers and Systems Engineering department. My working experience involved various jobs related to open source technologies, hardware platforms and high end power machines such as Training, Engineering, Implementation, Consultancy and Marketing. based on a number of platforms such as Red Hat, Fedora, Zenwalk, CentOS, Mandriva, Sabayon, Knopix, ArchLinux and

many others) over Linux flavor, and (BSD-Series, Sun Solaris, Oracle Solaris, HP-UX and many others) over Unix flavor.

Currently, I hold up to thirteen recognized certificates in Red Hat, Oracle Solaris, and Oracle SPARC. Lastly, I held the highest and powerful certificate all over the world "RHCA" and I was ranked number seven around Egypt and number fifteen around MENA.

I started my business in 2010 where I acted as a Co-Founder, UNIX/Linux Operations Manager and Trainer. Within a very short time, my start-up became like a beacon of Open Source in Mansoura Governorate - Egypt.



# DEVOPS WITH CHEF ON FREEBSD

## General description:

This training class teaches the tools, best practices and skills to automate your FreeBSD servers. Training will be loaded with practical real world tools and techniques. This training will send you back to work with immediately useful hands on experience to implement Devops in your IT projects.

Course launch date: **1st of December 2016, Self-Paced.**

Duration: 18 hrs.

## What will you learn?

- Learn what Devops is and its importance.
- Learn to leverage infrastructure automation using the leading configuration management tool: Chef.
- How it's changing the industry.
- Transform IT from an unpredictable environment to a stable, repeatable and scalable environment.
- Integrating configuration tools into the IT workflow.

## What skills will you gain?

- Configure development workstation.
- Understand the Chef architecture.
- Understanding different resources and automation.



**Module 1: Introduction to Devops and Chef.**

**Module 2: Understanding Chef resources.**

**Module 3: Setup the workstation as a local development environment.**

**Module 4: Bootstrap a node with Chef server.**

**Module 5: Get started with writing your first cookbook**

**Module 6: Recipes, Attributes, Metadata, Templates, etc.**

**Module 7: Roles, Environment, Search, etc.**

**Module 8: Using syntax and linting tools like: Foodcritic and Rubocop**

**Module 9: Writing unit test and integration tests.**

**There is only 50 seats available, so hurry up, book your seat as soon as possible.**

**You can find a full description of the course here:**

**<https://bsdmag.org/product/w05-devops-chef-freebsd/>**

**If you will have any questions regarding the course, don't hesitate to contact us at**

**[marta.ziemianowicz@bsdmag.org](mailto:marta.ziemianowicz@bsdmag.org)**



## MeetBSD 2016 Report

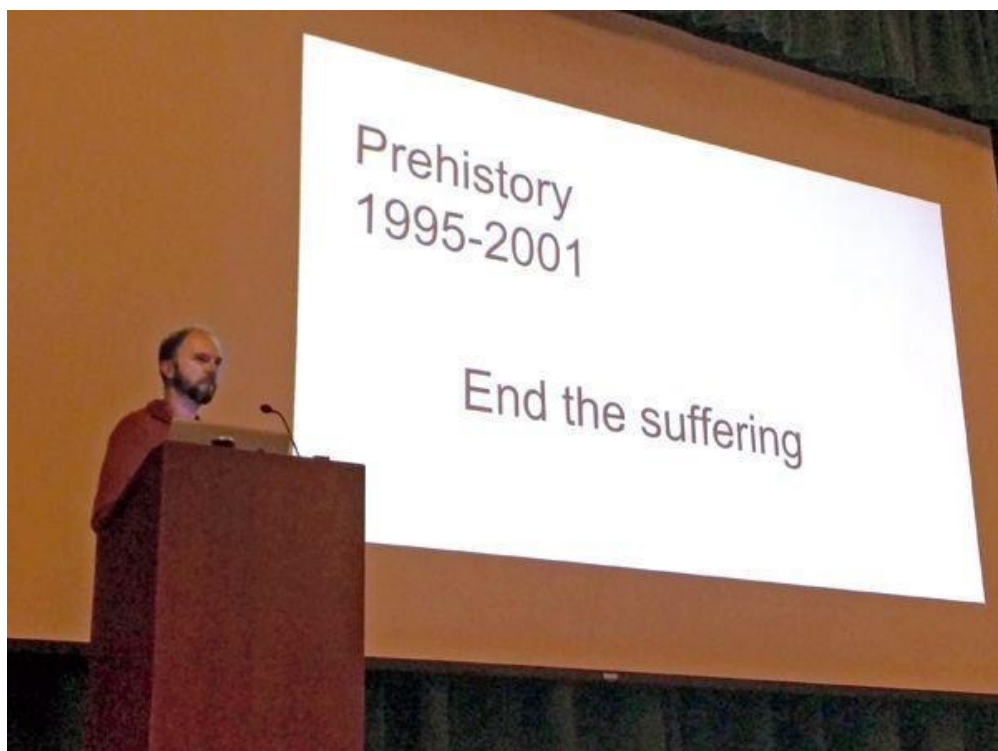
*by Michael Dexter*



MeetBSD 2016 wrapped up on November 12th at U.C. Berkeley in California, the birthplace of BSD Unix. Above all, the event reminded us all, the value of Bay Area BSDCons. Many local community members attended and even got a chance to present, who would otherwise not be able to make it to BSDCan, EuroBSDcon or AsiaBSDCon. One such presenter was ZFS co-developer, Matt Ahrens. He gave a great talk on

the history of ZFS and the problems that motivated its development. “You don’t have ‘RAM-adm’ to manage system memory when you add it... storage should be just as automatic.” This “local” factor also contributed to a packed FreeBSD Dev/Vendor Summit, the day before MeetBSD event. I have long denied this fact, but, Silicon Valley is still the heart of the tech industry and in many regards, the BSD community by extension.

The combined Developer and Vendor Summit included a traditional “Have/Need/Want” session and a presentation made by a team from Intel on FreeBSD NetBooting. The team found that by automating FreeBSD deployments, they could track down network driver bugs in hours rather than days or even weeks. The faster a system is configured, the faster they can diagnose it. This is a tribute to FreeBSD’s flexibility and its self-hosting nature. Other topics included “UFS in a ZFS era”, and the hallway track was excellent as always.







The first formal day of MeetBSD included a VNET/Jails talk by Devin Teske, a RISC-V talk by Krste Asanovic, a FreeBSD on Google Computer Engine talk by Sean Chittenden, Matt Ahrens' ZFS history lesson and my talk on bhyve. Devin and Sean presented compelling cases for using FreeBSD as a buzzword-free “cloud” platform and we learned how the RISC-V Open Source CPU design is coming along nicely. Matt's ZFS history talk included a photo of the moment when ZFS was first committed. My Monty

Python-themed “LIFE OF BHYVE” talk gave a similar history of bhyve and a list of upcoming features. Day one was wrapped up at the famous Hillside club near the campus and we all had a good time.

Day two of MeetBSD started with a history lesson by FreeBSD co-founder, Rod Grimes. His presentation detailed the emergence of FreeBSD from various 386BSD patch kits. Being the first FreeBSD release engineer, he made a point by referring us back then of \*not\* being the benevolent dictator of the project, but rather opted for a “core” team of leaders. FreeBSD co-founder, Jordan Hubbard, went on to give an update on FreeNAS 10 plus a tour of its virtualization features. FreeNAS 10 is proving quite capable and provides just about the easiest way to try the bhyve hypervisor. PC-BSD founder and BSDNow co-host, Kris Moore, then gave a talk about the evolution of PC-BSD into TrueOS and demonstrated its latest features such as “SysAdm” and the Lumina desktop. The last formal session was a panel discussion on OpenZFS which included FreeBSD ZFS importer, Pawel Dawidek, iXsystems Storage Architect, Josh Paetzel, ZFS Book co-author and



BSDNow co-host, Allan Jude and BSDCan organizer, Dan Langille. It's obvious that MeetBSD event was a who's who of BSD Unix with an emphasis on FreeBSD, thanks to iXsystems' strong role in that community.

FreeBSD release engineer, he made a point by referring us back then of \*not\* being the benevolent dictator of the project, but rather opted for a "core" team of leaders. FreeBSD co-founder, Jordan Hubbard, went on to give an update on FreeNAS 10 plus a tour of its virtualization features. FreeNAS 10 is proving quite capable and provides just about the easiest way to try the bhyve hypervisor. PC-BSD founder and BSDNow co-host, Kris Moore, then gave a talk about the evolution of PC-BSD into TrueOS and demonstrated its latest features such as "SysAdm" and the Lumina desktop. The last formal session was a panel discussion on OpenZFS which included FreeBSD ZFS importer, Pawel Dawidek, iXsystems Storage Architect, Josh Paetzel, ZFS Book co-author and BSDNow co-host, Allan Jude and BSDCan organizer, Dan Langille. It's obvious that MeetBSD event was a who's who of BSD Unix with an emphasis on FreeBSD, thanks to iXsystems' strong role in that community.

I truly appreciate the laid-back nature of MeetBSD because it shifts majority of the technical discussions to the hallway track, which is perfect in a sunny Berkeley. We didn't have a Computer Science Research Group tour. However, various people reported of meetings and meals they had with people from "back in the day." Moreover, I was delighted that authorities like Rod Grimes joined the discussion once again. Both FreeBSD-centric and MeetBSD attracted a group of OpenBSD users and developers, plus a few representatives of NetBSD and DragonFly BSD. Being at Berkeley turned out to be a time well spent, discussions even touched on DEC PDP-11 and Apollo topics! I look forward to MeetBSD 2018 and vBSDcon 2017 in the interim. See you there!

### About the Author:

Michael Dexter, Senior Analyst  
iXsystems, Inc.

**Over the weekend, Tesco Bank suspended online transactions after an attacker gained access to over 20,000 accounts, with money being withdrawn fraudulently in some cases. In another security incident, the City of El Paso was the victim of CEO fraud worth over \$3.2 million. What implications does this paradigm shift towards online crime have for us as a society?**

***by Rob Somerville***

To those who are closely involved in IT security, it will come as no surprise that the sheer scale of online fraud continues to rise, with larger and larger amounts of money involved and the tentacles of the black hats reaching every section of society, from the individual to the multi-national. As far as the banking sector is concerned, there is more than a smattering of irony here if we are to look back in history to the 1970's. At that time (certainly in the UK at least), there was an epidemic of armed robberies on banks and large employers with cash payrolls, which led to a number of major changes to counteract the threat. The banking sector spent millions on additional security, installing state of the art alarm systems, bullet proof glass and automated shutters that descended from the ceiling in the blink of an eye. The security industry enjoyed a growth phase, and the judiciary handed out harsh sentences (sometimes longer than that for murder) as a deterrent to any that were foolish enough to carry out such crimes. However, it was not until the widespread adoption of salary payments being paid directly into em-

ployee's bank accounts did armed robbery go out of vogue. Very few employers nowadays (if any) will pay an employee in cash. The individual now bears the physical risk of visiting that ATM on the dark street in that shady part of town. The irony is, almost 50 years later, the criminals are back with a vengeance but instead of shotguns and stocking masks, the tools are security scanners, proxies, social engineering and scripts. The drive for efficiency and convenience has come full circle and the banks, the business and the individual are once again a legitimate mass target.

Financial or "white collar crime" has always existed, of course, but the shift of banking and electronic fund transfer to the global public world of the Internet has been a game changer in increasing the risk footprint. In the pre-internet days, fund transfer was via dedicated secure lines and the banks had substantial control as the customer was required to register their telephone number, otherwise they could not log onto the system, and security was also enhanced with one



time passwords. On the downside, customer traffic was via the Plain Old Telephone System, the traffic was not encrypted and in the days prior to dedicated software for this purpose, all the customer needed was a suitable modem and some terminal emulation software, the more advanced versions which would allow you to capture then entire session to a text file, a major security risk. The chances of a criminal gang from a foreign country attempting to empty your bank account was negligible, especially as the audio quality of international telephone calls would mean a large number of CRC data errors, and the cost of phoning the UK prohibitive.

While automation and the Internet has played a part in making life easier for the masses, the same applies to the cyber-criminal. The exact methodology used to gain access to the Tesco accounts has not been disclosed, but this could have been any number of ways. A compromised mobile application, an attack on the website itself, a staff member or poor password hygiene on the part of the customer are all possible vectors. Coupled with even a modest bot-net, the opportunity for discovering weaknesses in the system improves the odds considerably without resorting to the risky, time intensive, brute force attack, which hopefully would alert the victim to suspicious activity.

Once again, it is ironic that the methods used in the 1970's will be the most successful in defeating the criminal, albeit with a modern technological twist. The difference this time round is that both the organisation and the customer need to take steps to eliminate the risks; if either side is lax in addressing security, this

compromises the other party. The picture is further complicated as other parties need to be security conscious as well. The customer and organisation can take every reasonable precaution, but this will be negated if the device in question used to access the service has poor long term patch support from the manufacturer or is vulnerable to exploit. In a consumer driven marketplace, it is unlikely that this will be a concern to many outside the IT sector.

The implications for both organisation and the individual will be many, if indeed the whole issue of cyber security is to be taken seriously. In the instance of CEO fraud, apart from the obvious tightening of internal procedures and improving email security by mitigating phishing attacks, education, effective communication and culture are also critical factors in reducing risk. This will potentially also impact suppliers, as they too need to be involved.

As far as banking fraud is concerned, unlike the 1970's, we do not have the option of changing the method of payment. Much further investment is required in securing and testing websites, applications, devices and procedures. Moving from the simple username / password / passphrase combination to two or possibly even three factor authentication would be a start, even returning to the days of an additional one-time disposable key via key fob. While these measures will improve security dramatically, there will always be a downside in that not only are they costly, but may be a barrier to entry and convenience is often preferred over integrity.

Increased auditing, limiting access to licensed or approved devices / software and more strin-



gent vetting and checking of staff and business partners are just the start. In the arms race of business versus the hacker, security does not come cheap, either in terms of financial cost or time to test adequately and highlight vulnerabilities. We may even return to the days of the hardware dongle, and mandatory custom software for banking provided by the banks themselves. While this might seem unthinkable in the age of the browser, it would be a natural evolution to move towards closed mobile / tablet applications if major improvements in security can be demonstrated.

Irrespective of how these problems are addressed, we can be sure of two things. Once the business sector grasps the nettle of security fully, the pain will be expressed in increased cost and more rigorous examination of individuals both inside and outside the organisation. The other option is to truly return to the 1970's – El Paso is currently using old fashioned paper invoicing until they get to the bottom of the matter.